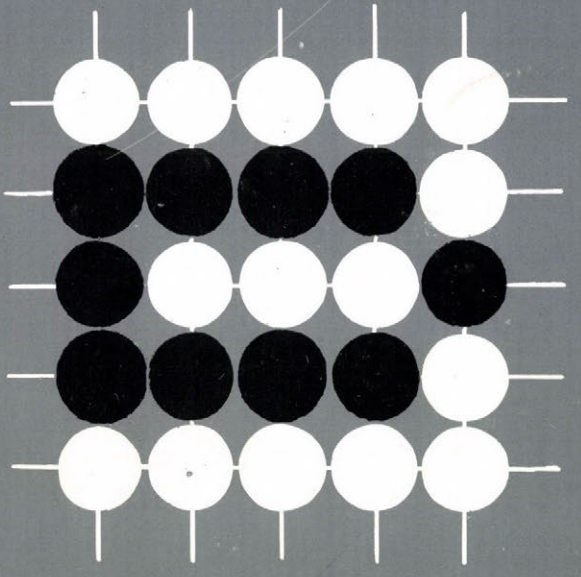
Közlemények

37/1987

MTA Számítástechnikai és Automatizálási Kutató Intézet

Budapest



Magyar Tudományos Akadémia
Számítástechnikai és Automatizálási Kutató Intézete
Computer and Automation Institute, Hungarian Academy of Sciences

K Ö Z L E M É N Y E K

T R A N S A C T I O N S

37/1987

Felelős kiadó:

DR. KEVICZKY LÁSZLÓ

ISBN 963 311 243 5
ISSN 0133-7459

technografik ksz.

Készült az OMIKK házinyomdájában
Budapest I., Gyorskocsi u. 5-7.
Felelős vezető: Tóth Károly

CONTENTS

MTA SZTAKI Közlemények 37/1987 pp. 5-20

ON MOVING AVERAGE COMPOSITION AND PREDICTIVE DECONVOLUTION OF LONG-RUN STATIONARY TIME SERIES

PHAN DANG CAU

Computer and Automation Institute
Hungarian Academy of Sciences

ABSTRACT

The notion of the time mean or time average of time series is widely used in the literature on time series analysis. Analogously, the notion of the time autocorrelation function of the time series x_n can be defined as the limit of the sum

$$\frac{1}{N}\sum_{n=0}^{N-1} x_{n+s}x_n \quad , \; s = 0,1,\ldots \text{ when } N \text{ tends to } \infty$$

However, in standard literature on time series analysis only the case when the time mean and the time autocorrelation function coincide with the mean Ex_n and the autocorrelation function $\varphi_s = Ex_{n+s}x_n$ is investigated. In [7] we have shown that the seismic signals are not always stationary in usual sense, but they are stationary in slight different sense, i.e. they are so-called long-run stationary autoregressive time series. We have also shown in [7] that for long-run stationary autoregressive time series, the time autocorrelation function is not the same as the autocorrelation function. However, in the predictive deconvolution problem of this time series , instead of the autocorrelation function, the time correlation function plays an important role.

By this reason, in this paper we propose to consider the time mean and the time correlation function as independent notions. / They coinside with the mean and the autocorrelation function when the process x_n is ergodic and stationary, only/.

We also introduce the notion of a long-run stationary process, and investigate the moving average composition and predictive deconvolution of this process.

§.1. DEFINITIONS AND NOTIONS

Definiton 1: The time series x_n is said to have the time mean $M\{x_n\}$ in the sense of convergence almost everywhere / in probability/ if the sum

$$\frac{1}{N}\sum_{n=m+1}^{m+N} x_n$$

converges to $M\{x_n\}$ almost everywhere / in probability /

Definition 2: The time series x_n is said to have the time autocorrelation function $g_x(s)$ in the sense of convergence almost everywhere / in probability/ if for $s=\ldots-1,0,1,2,\ldots$ the sums

$$\frac{1}{N}\sum_{n=m+1}^{m+N} x_{n+s}x_n$$

converge to $g_x(s)$ almost everywhere / in probability/

Remark: The following example shows that the time mean of some time series may be the same as the mean, while the time autocorrelation function may be different from the autocorrelation function.

Example 1: Let $x_n=\cos nW$ $n=1,2,\ldots$ where W is a uniformly distributed random variable on $[0,2\pi]$. Then we can see

$$M\{x_n\} = Ex_n = 0$$

$$g_x(1) = \frac{1}{2}\cos W \neq 0 = Ex_{n+1}x_n \quad a.s.$$

Definition 3: Let x_n and y_n be two arbitrary time series. If for $s=\ldots-1,0,1,2,\ldots$ the sums

$$\frac{1}{N}\sum_{n=m+1}^{m+N} x_{n+s}y_n$$

converge in a given sense to $g_{xy}(s)$ then $g_{xy}(s)$ is called the time crosscorrelation function between x_n and y_n in the given sense of convergence .

Definition 4: If the time series x_n and y_n have the time cross-correlation function and $g_{xy}(s) = 0$, $s = 0,1,\ldots$ then we call they are time-uncorrelated.

Definition 5: If for the time series u_n, $n = 0,1,\ldots$

$$g_u(0) = \sigma^2 > 0 \quad , \quad g_u(s) = 0 \quad , \quad s = 1,2,3,\ldots$$

where σ^2 may be random variable, then u_n is said to be time-uncorrelated / in the given sense of convergence /.

Definition 6: If the time series u_n is time-uncorrelated and has the time mean 0, then we call u_n a long-run white noise / in the given sense of convergence /

Definition 7: The time series x_n is called a long-run stationary time series / in the given sense of convergence / if it has the time mean and the time correlation function / in the given sense of convergence /

We would like to notice that the time mean and the time auto-correlation function may be random variables.

From now on, if we do not say else, we always mean that the above definitions are given in the sence of convergence almost everywhere .

§.2. PROPERTIES OF THE TIME MEAN AND THE TIME -AUTOCORRELATION FUCTION

1. If x_n and y_n are long-run stationary time series and there exists the time crosscorrelation function between them, then $z_n = \alpha x_n + \beta y_n$ has the same property and

$$M\{z_n\} = \alpha M\{x_n\} + \beta M\{y_n\} \qquad a.s \qquad (1)$$

$$g_z(s) = \alpha^2 g_x(s) + \alpha\beta\, g_{xy}(s) + \alpha\beta\, g_{yx}(s) + \beta^2 g_y(s) \quad a.s \qquad (2)$$

Thus if x_n and y_n are time-uncorrelated then

$$g_z(s) = \alpha^2 g_x(s) + \beta^2 g_y(s) \qquad (3)$$

The proof is straightforward.

2. $g_{xy}(s) = g_{yx}(-s)$ a.s. (4)

3. $|g_x(s)| \leq g_x(0)$ a.s. (5)

Proof: It is obvious that

$$0 \leq M\left\{(x_{n+s} \pm x_n)^2\right\} = M\left\{x_{n+s}^2 \pm 2x_{n+s}x_n + x_n^2\right\} = 2g_x(0) \pm 2g_x(s)$$

which proves (5) .

4. $g_{xy}^2(s) \leq g_x(0)\, g_y(0)$ (6)

Proof: We can see

$$0 \leq M\left\{(ax_{n+s} + by_n)^2\right\} = M\left\{a^2x_{n+s}^2 + 2abx_{n+s}y_n + b^2y_n^2\right\} =$$

$$= a^2g_x(0) + 2ab\,g_{xy}(s) + b^2g_y(0)$$

The above quadratic form is nonnegative for any a, b therefore its discriminant is nonnegative. Thus (6) is proved.

5. $g_x(s)$ is nonnegative definite a.s. (7)

Proof: Let $a_1, a_2, \ldots, a_M$ be arbitrary values, then we have

$$\sum_{s,\ell=1}^{M} g_x(s-\ell)\, a_s a_\ell = \sum_{s,\ell=1}^{M} \lim_{N\to\infty} \frac{1}{N} \sum_{n=m+1}^{m+N} x_{n+s}\, x_{n+\ell}\, a_s a_\ell =$$

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=m+1}^{m+N} \sum_{s,\ell=1}^{M} x_{n+s}\, x_{n+\ell}\, a_s a_\ell = \lim_{N\to\infty} \frac{1}{N} \sum_{n=m+1}^{m+N} \left(\sum_{s=1}^{M} x_{n+s} a_s\right)^2 \geq 0$$

Thus we have (7) .

6. There exists a random function $F(\lambda)$, which is a spectral distribution function a.s. and

$$g_x(s) = \int_{-0.5}^{0.5} e^{2\pi i \lambda s}\, dF(\lambda) \qquad \text{a.s.} \tag{8}$$

We can see that (8) is an immediate consequence of (7) .
We call $F(\lambda)$ the time-spectral distribution function of the time series x_n .
If $F(\lambda)$ is absolutely continuous a.s. then $f(\lambda) = F'(\lambda)$ is called the time-spectral density function of x_n .

7. $$\int_{\mu_1}^{\mu_2} \frac{1}{N}\left|\sum_{k=0}^{N-1} x_k e^{-i2\pi\lambda k}\right|^2 d\lambda = F(\mu_2) - F(\mu_1) \quad a.s \qquad (9)$$

where the random variables μ_1, μ_2 are the continuity points of $F(\lambda)$.

8. Linear operator on long-run stationary time series.
Suppose the sequence h_τ , $\tau = \ldots -1, 0, 1, \ldots$ satisfies the following conditions:

$$\sum_{\tau=-\infty}^{\infty} |h_\tau| < \infty \quad , \quad \sum_{\tau=-\infty}^{\infty} h_\tau^2 < \infty \quad . \qquad (10)$$

and x_n is arbitrary long-run stationary time series then

$$y_n = \sum_{\tau=-\infty}^{\infty} h_\tau x_{n-\tau} \qquad (11)$$

is also long-run stationary time series with

$$M\{y_n\} = M\{x_n\} \sum_{\tau=-\infty}^{\infty} h_\tau \qquad (12)$$

$$g_y(s) = \sum_{\tau=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} h_\tau h_k g_x(s-\tau+k) \qquad (13)$$

Proof: The proof of (12) is straightforward . To prove (13) let us consider

$$\frac{1}{N}\sum_{n=m+1}^{m+N} y_{n+s} y_n = \sum_{\tau=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} h_\tau h_k \frac{1}{N} \sum_{n=m+1}^{m+N} x_{n+s-\tau} x_{n-k}$$

from which

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=m+1}^{m+N} y_{n+s} y_n = \sum_{\tau=-\infty}^{\infty} \sum_{k=-\infty}^{\infty} h_\tau h_k g_x(s-\tau+k)$$

We notice that by (5) the series on the right hand side of (13) converges.

§.3. MOVING AVERAGE COMPOSITION OF LONG-RUN STATIONARY TIME SERIES

Theorem 1: a. If the time series x_n has the form

$$x_n = \sum_{s=0}^{\infty} b_s u_{n-s} \qquad (14)$$

where $b_0, b_1, b_2, \ldots$ are random variables for which

$$\sum_{s=0}^{\infty} b_s^2 < \infty \qquad (15)$$

and u_n is time-uncorrelated then x_n has the time spectral density function $f(\lambda)$ and

$$\int_{-0.5}^{0.5} \ln f(\lambda) d\lambda > -\infty \qquad a.s \tag{16}$$

b. Conversely, if the time series x_n has the time spectral density function $f(\lambda)$, for which (16) holds then x_n can be written in the form (14), where u_n is time-uncorrelated. If u_n is a long-run white noise, then x_n is a long-run stationary process with $M\{x_n\} = 0$.

Proof: a. By (13) we have

$$g_x(s) = \sum_{r=0}^{\infty} \sum_{k=0}^{\infty} b_r b_k g_u(s - r + k) = \sigma^2 \sum_{n=0}^{\infty} b_{n+s} b_n$$

Thus

$$f(\lambda) = \sigma^2 \left| \sum_{n=0}^{\infty} b_n e^{i 2\pi n \lambda} \right|^2$$

i.e. the problem is reduced to the theory of analytic functions / H^2 classes /

b. For $f(\lambda)$ satisfying (16) we can write

$$f(\lambda) = \sigma^2 \left| \sum_{n=0}^{\infty} b_n e^{i 2\pi n \lambda} \right|^2 \qquad a.s$$

where $b_0 = 1$ and $\sum_{n=0}^{\infty} b_n z^n \neq 0$ for $|z| \leq 1$.
By the definition of the time specral density we have

$$g_x(s) = \sigma^2 \sum_{n=0}^{\infty} b_{n+s} b_n$$

Set

$$A(z) = a_0 + a_1 z + a_2 z^2 + \cdots = \frac{1}{\sum_{n=0}^{\infty} b_n z^n}$$

and

$$u_n = \sum_{s=0}^{n} a_s x_{n-s} \qquad , n = 0, 1, 2, \cdots$$

Then

$$x_n = \sum_{s=0}^{n} b_s u_{n-s}$$

Now we prove that u_n is time-uncorrelated.

To do this let us consider

$$q_u(s) = \sum_{i=0}^{\infty}\sum_{k=0}^{\infty} a_i a_k q_x(s-i+k) =$$

$$= \sigma^2 \sum_{i=0}^{\infty}\sum_{k=0}^{\infty} a_i a_k \sum_{n=0}^{\infty} b_{n+s-i} b_{n-k} =$$

$$= \sigma^2 \sum_{n=0}^{\infty}\sum_{i=0}^{\infty} a_i b_{n+s-i} \sum_{k=0}^{\infty} a_k b_{n-k} =$$

$$= \sigma^2 \sum_{n=0}^{\infty} d_{n+s} d_n = \begin{cases} \sigma^2 & s=0 \\ 0 & s\neq 0 \end{cases}$$

where $D(z) = 1 + d_1 z + d_2 z^2 + \cdots = A(z)B(z) = 1$

Thus the theorem is proved.

§.4. PREDICTIVE DECONVOLUTION OF AUTOREGRESSIVE TIME SERIES WITH RANDOM COEFFICIENTS

In [7] we have investigated the predictive deconvolution problem of long-run stationary autoregressive time series x_n which satisfies the following equation:

$$x_n + a_1 x_{n-1} + a_2 x_{n-2} + \cdots + a_p x_{n-p} = u_n \quad , n = 0,1,2,\cdots \qquad (17)$$

where the u_n's are independent random variables with mean 0 and variance $Eu_n^2 = \sigma_n^2$ and

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} Eu_n^2 = \sigma^2 > 0$$

However, it is reasonable to assume that among the u_n's there may be dependent random variables and variables having non-zero mean. Similarly, the coefficients $a_1, a_2, \ldots, a_p$ may depend on experiment, i.e. they may be random variables. In this section we investigate this case.

Theorem 2: Let x_n be a process satisfying the equation

$$x_n + \alpha_1 x_{n-1} + \cdots + \alpha_p x_{n-p} = u_n \quad n = 0,1,2,\cdots \qquad (18)$$

where $\alpha_1, \alpha_2, \cdots, \alpha_p$ are random variables and the following three conditions:

a. Let $\mathcal{N}$ denote the set of non-negative integers, and let H be a subset of $\mathcal{N}$ such that

$$\lim_{N\to\infty} \frac{1}{N} \sum_{\substack{n\le N-1 \\ n,n+s\in H}} u_{n+s}u_n = 0 \quad a.s \tag{19}$$

b. The u_n's, $n\in\mathcal{N}\setminus H$ are independent variables with mean 0 and

$$E|u_n|^{2+\varepsilon_0} < K < \infty \quad , \quad \lim_{N\to\infty} \frac{1}{N} \sum_{\substack{n\le N-1 \\ n\in\mathcal{N}\setminus H}} Eu_n^2 = \sigma^2 > 0 \tag{20}$$

c. $$1 + \alpha_1 z + \alpha_2 z^2 + \dots + \alpha_p z^p \neq 0 \quad \text{for} \quad z \le 1 \tag{21}$$

Then x_n is a long-run stationary process with $M\{x_n\} = 0$. The matrix

$$\begin{pmatrix} g_x(0) & g_x(1) & \cdots & g_x(p-1) \\ g_x(1) & g_x(0) & \cdots & g_x(p-2) \\ \vdots & & & \\ g_x(p-1) & g_x(p-2) & \cdots & g_x(0) \end{pmatrix} \tag{22}$$

is positive definite a.s. and we have

$$\begin{pmatrix} g_x(0) & g_x(1) & \cdots & g_x(p-1) \\ g_x(1) & g_x(0) & \cdots & g_x(p-2) \\ \vdots & & & \\ g_x(p-1) & g_x(p-2) & \cdots & g_x(0) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_p \end{pmatrix} = - \begin{pmatrix} g_x(1) \\ g_x(2) \\ \vdots \\ g_x(p) \end{pmatrix} \tag{23}$$

Proof: Let

$$v_n = \begin{cases} 0 & n \in H \\ u_n & n \notin H \end{cases}$$

then the sequence v_n, $n=0,1,2,\dots$ satisfies the conditions of the theorem in [7], hence we have

$$g_u(s) = g_v(s) = \begin{cases} \sigma^2 & s = 0 \\ 0 & s \neq 0 \end{cases}$$

Now we can write

$$x_n = \sum_{s=0}^{n} \beta_s u_{n-s} \tag{24}$$

where

$$1 + \beta_1 z + \beta_2 z^2 + \dots = \frac{1}{1 + \alpha_1 z + \dots + \alpha_p z^p} \tag{24'}$$

By (18) and (24)

$$\sum_{\tau=0}^{p} \alpha_\tau x_{n-\tau} x_{n-k} = \sum_{s=0}^{n-k} \beta_s u_n u_{n-k-s}$$

from which we have

$$\sum_{\tau=0}^{p} \alpha_\tau g_x(k-\tau) = \sum_{s=0}^{\infty} \beta_s g_u(s+k)$$

For $k= 1,2,\ldots,p$ we get (24) .
Notice that

$$g_x(s) = \sum_{n=0}^{\infty} \beta_{n+s} \beta_n$$

then it is well known that (21) and (24') provide the positivity of the matrix (23) . The proof is complete.

Remark: 1. If H is finite, then (19) is satisfied.
2. If $|u_n| < K$, $n \in H$ and

$$\lim_{N\to\infty} \frac{1}{N} \sum_{n=0}^{N-1} \chi_H(n) = 0 \qquad \text{where } \chi_H(n) = \begin{cases} 1 & n \in H \\ 0 & n \notin H \end{cases}$$

then (19) is also satisfied .
3. If we know $g_x(\tau)$, $\tau = 0,\ldots,p$ then α_τ's are determined by (23) . In practice we can estimate $g_x(\tau)$ by

$$r_\tau = \frac{1}{N} \sum_{n=0}^{N-1} x_{n+\tau} x_n$$

and then $\alpha_1, \alpha_2, \ldots, \alpha_p$ are estimated by

$$\begin{pmatrix} r_0 & r_1 & \cdots & r_{p-1} \\ r_1 & r_0 & \cdots & r_{p-2} \\ \vdots & & & \\ r_{p-1} & r_{p-2} & \cdots & r_0 \end{pmatrix} \begin{pmatrix} \hat{\alpha}_1 \\ \hat{\alpha}_2 \\ \vdots \\ \hat{\alpha}_p \end{pmatrix} = - \begin{pmatrix} r_1 \\ r_2 \\ \vdots \\ r_p \end{pmatrix} \tag{25}$$

Therefore we can estimate u_n by $\hat{u}_n = x_n + \hat{\alpha}_1 x_{n-1} + \cdots + \hat{\alpha}_p x_{n-p}$
and we have

$$\lim_{N\to\infty} \hat{u}_n = u_n \qquad a.s.$$

Numerical example:
Example 2: Let x_n be a long-run stationary process defined by

$$x_n + \alpha x_{n-1} = u_n \qquad \alpha = 0.5 \ , \ n = 0, 1, \cdots, 499 \tag{26}$$

$$u_n = \begin{cases} W_n & v_n \leq 0.5 \\ 0 & v_n > 0.5 \end{cases} \qquad n \neq 4,5,6,9$$

$$u_4 = u_5 = u_6 = u_9 = v \tag{27}$$

where W_n's are independently, normally r.v.'s with mean 0 and variance 1, v_n's and v are independenly , uniformly r.v.'s on [0,1] and independent of W_n's .
Thus

$$E u_4 = E u_5 = E u_6 = E u_9 = 0.5 \neq 0 \ , \quad \lim_{N \to \infty} \frac{1}{N} \sum_{n=0}^{N-1} E u_n^2 = 0.5$$

By (25) we get the estimate $\hat{\alpha}$ =0.4941 . In table 1 the first 30 values of u_n and its estimate are printed.

Table 1: Long-run white noise defined by (27) and its estimate

n	long-run white noise	estimated long-run white noise	n	long-run white noise	estimated long-run white noise
0	0.1346	0.1346	15	0.0	-0.0061
1	0.2081	0.2073	16	0.0	0.0031
2	-0.1960	-0.1968	17	0.0	-0.0015
3	0.0	0.0016	18	0.0	0.0008
4	0.1527	0.1519	19	0.7724	0.7720
5	0.1527	0.1522	20	0.6930	0.6886
6	0.1527	0.1521	21	0.6723	0.6704
7	0.0	-0.0006	22	0.0541	0.0510
8	0.0	0.0003	23	0.0	0.0012
9	0.1527	0.1526	24	0.0	-0.0006
10	0.0	-0.0008	25	0.1007	0.1010
11	0.5493	0.5497	26	0.0	-0.0007
12	0.0	-0.0035	27	0.0	0.0004
13	0.0	0.0017	28	0.3580	0.3579
14	1.1048	1.1040	29	-0.0502	-0.0522

We can see that u_4, u_5, u_6 and u_9 are well estimated inspite of their dependence and the fact that they have non-zero mean.

§.5. PREDICTIVE DECONVOLUTION OF ARI-TIME SERIES

It is reasonable to think that in some applications

$$A(z) = 1 + \alpha_1 z + \cdots + \alpha_p z^p \neq 0 \qquad \text{for} \quad |z| \leq 1$$

there may be some roots which are close to the unit circle. In this section we investigate the predictive deconvolution problem of the limiting case, i.e., the case in which there are some roots on the unit circle.

It is easy to prove the following theorem:

Theorem 3: Suppose the process x_n satisfies the conditions (18), (19), (20) and the Z-transform can be written in the form

$$A(z) = (1-z)^d \left(1 + \gamma_1 z + \cdots + \gamma_q z^q\right) \tag{28}$$

where $q + d = p$

$$1 + \gamma_1 z + \cdots + \gamma_q z^q \neq 0 \qquad \text{for} \quad |z| \leq 1 \tag{29}$$

Let $y_n = \Delta^d x_n$ where $\Delta x_n = x_{n+1} - x_n$

Then y_n is a long-run stationary process with $M\{y_n\} = 0$.

The matrix

$$\begin{pmatrix} g_y(0) & g_y(1) & \cdots & g_y(q-1) \\ \vdots & & & \\ g_y(q-1) & g_y(q-2) & \cdots & g_y(0) \end{pmatrix}$$

is positive definite and we have

$$\begin{pmatrix} g_y(0) & g_y(1) & \cdots & g_y(q-1) \\ g_y(1) & g_y(0) & \cdots & g_y(q-2) \\ \vdots & & & \\ g_y(q-1) & g_y(q-2) & \cdots & g_y(0) \end{pmatrix} \begin{pmatrix} \gamma_1 \\ \gamma_2 \\ \vdots \\ \gamma_q \end{pmatrix} = - \begin{pmatrix} g_y(1) \\ g_y(2) \\ \vdots \\ g_y(q) \end{pmatrix} \tag{30}$$

In practice the predictive deconvolution of ARI - time series can be done as follows :

1. Compute $y_n = \Delta^d x_n$
2. Estimate $g_y(\tau)$ by $r_y(\tau)$
3. Solve the normal equations to get the estimates $\hat{\gamma}_1, \hat{\gamma}_2, \ldots, \hat{\gamma}_q$
4. Estimate $\alpha_1, \alpha_2, \ldots, \alpha_p$ by

$$1 + \hat{\alpha}_1 z + \cdots + \hat{\alpha}_p z^p = (1-z)^d \left(1 + \hat{\gamma}_1 z + \cdots + \hat{\gamma}_q z^q\right)$$

5. Estimate u_n's by

$$\hat{u}_n = x_n + \hat{\alpha}_1 x_{n-1} + \cdots + \hat{\alpha}_p x_{n-p}$$

Then we have

$$\lim_{N \to \infty} \hat{u}_n = u_n \quad a.s$$

Numerical example:

Example 3: Let x_n be ARI-time series defined by

$$x_n + \alpha_1 x_{n-1} + \alpha_2 x_{n-2} = u_n \quad , \alpha_1 = -0.5, \alpha_2 = -0.5, n = 0, \ldots, 499 \qquad (31)$$

where u_n is defined by (27) .
Then we get the estimates $\hat{\alpha}_1 = -0.5058$, $\hat{\alpha}_2 = -0.4942$. In table 2 the first 20 values of u_n and its estimate are printed.

Table 2: Long-run white noise defined by (27) and its estimate

n	long-run white noise	estimated long-run white noise	n	long-run white noise	estimated long-run white noise
0	0.1346	0.1346	10	0.0	-0.0008
1	0.2081	0.2073	11	0.5493	0.5497
2	-0.1960	-0.1968	12	0.0	-0.0034
3	0.0	0.0016	13	0.0	0.0017
4	0.1527	0.1519	14	1.1048	1.1040
5	0.1527	0.1522	15	0.0	-0.0060
6	0.1527	0.1521	16	0.0	0.0030
7	0,0	-0.0006	17	0.0	-0.0015
8	0.0	0.0003	18	0.0	0.0008
9	0.1527	0.1526	19	0.7724	0.7720

§.6. THE DETERMINATION OF THE ORDER OF ARI-TIME SERIES

As we have seen in the previous section, in the predictive deconvolution of ARI-time series we need to know the parameters d and q . However, in the majority of applications the observations x_ns are available only , thus we have to estimate the parameters d and q . In [7] we have shown that the usual methods for testing statistical hypotheses cannot be always applied to the long-run stationary processes. Here we propose to estimate d and q as follows:

1. For k=0,1,2,... compute $y_n = \Delta^k x_n$

If for some k_0 and for fixed $s = 0,1,2,\cdots$ the average

$$\frac{1}{N}\sum_{n=m+1}^{m+N} y_{n+s} y_n$$

stabilizes for large N then we estimate d by k_0 / It means that the process $y_n = \Delta^{k_0} x_n$ is long-run stationary time series /

2. For $j = 1,2,\cdots$ we solve the equations

$$\begin{pmatrix} r_y(0) & r_y(1) & \cdots & r_y(j-1) \\ r_y(1) & r(0) & \cdots & r_y(j-2) \\ \vdots & & & \\ r_y(j-1) & r_y(j-2) & \cdots & r_y(0) \end{pmatrix} \begin{pmatrix} \hat{\alpha}_1^j \\ \hat{\alpha}_2^j \\ \vdots \\ \hat{\alpha}_j^j \end{pmatrix} = - \begin{pmatrix} r_y(1) \\ r_y(2) \\ \vdots \\ r_y(j) \end{pmatrix}$$

Then we compute

$$\hat{u}_n^j = 1 + \hat{\alpha}_1^j x_{n-1} + \cdots + \hat{\alpha}_{j+d}^j x_{n-j-d}$$

$$c_{\hat{u}}^j(s) = \frac{1}{N}\sum_{n=0}^{N-1} \hat{u}_{n+s}^j \hat{u}_n^j$$

If for some j_0 we have

$$c_{\hat{u}}^{j_0}(0) = \sigma^2 \quad , \quad c_{\hat{u}}^{j_0}(s) \approx 0 \ , \ s = 1,2,\cdots$$

i.e. $c_{\hat{u}}^{j_0}(s)$, $s = 1,2,\cdots$ are negligible compared with $c_{\hat{u}}^{j_0}(0)$ then we estimate q by j_0 .

Acknowledgement: The author is very grateful to his teacher, Dr. József Tomkó and Dr. András Krámli for their guidance, encouragement and many valuable discussions during the preparation of this paper.

REFERENCES

[1] Anderson,T.W /1972/ Statistical Analysis of Time series John Wiley and sons.

[2] Arató,M., Benczur,A., Krámli,A., Pergel,J. /1974/ Statistical problems of elementary Gaussian process, I. stochastic process / MTA SZTAKI Tanulmányok 22/1974/

[3] Doob,J.L /1953/ Stochastic processes, John Wiley and sons, Inc., New York.

[4] Gihman I.I, SZkorohod A.V /1975/ Bevezetés a sztochasztikus folyamatok elméletébe, Müszaki könyvkiadó, Budapest.

[5] Krámli,A. and Pergel,J. /1974/ The connection between Gaussian Markov processes and Autoregressive Moving Average processes, / MTA SZTAKI Közlemények 13/1974 /

[6] Meskó,A /1984/ Digital filtering applications in Geophysical Exploration for oil, Akadémiai Kiadó, Budapest .

[7] Phan Dang Cau /1987/ On predictive deconvolution of long-run stationary time series, MTA SZTAKI Közlemények 36/1986 / To appear /

[8] Robinson,E.A /1981/ Time series analysis and applications, Houston, Goose Pond Press.

On moving average composition and predictive deconvolution of long-run stationary time series

Phan Dang Cau

Summary

The notions of empirical and long-run stationary processes respectively, are introduced.
It is proposed to consider the time mean and the time correlation functions as independent notions /they coincide with the mean and the autocorrelation function when the process is ergodic and stationary, only/.
The moving average composition and predictive deconvolution of the long-run process are investigated.

Empirikusan stacionárius folyamat mozgóátlag felbontása és prediktiv dekonvolúciója

Phan Dang Cau

Összefoglaló

A cikkben a szerző bevezeti az empirikusan stacionárius folyamat fogalmát, vizsgálja e folyamat mozgóátlag felbontását, prediktiv dekonvolúcióját.

MTA SZTAKI Közlemények 37/1987 pp. 21-61

A MATHEMATICAL THEORY OF PROCESSORS I.

Ernö Gesztelyi

DEPT. OF COMPUTER SCIENCE
LAJOS KOSSUTH UNIVERSITY, DEBRECEN

With the development of computer technics, many new concepts came to light and became generally known, such as Read Only and Random Access Memory, Microprocessor, Interface, etc. The question on the function and role of these devices raises naturally.

The behaviour of these devices may be described at different levels. At a low level, which will be called the hardware level, one deals with transistors, capacitors, resistors and with the behaviour characteristics of integrated-circuit /IC/ chips, etc. At a higher level, which will be called the software level, we could describe logical circuits, diagrams showing the hardware organization of a computer system. At this level one gives the functional description of an IC and the instruction set, etc. Both of these description levels are used widely in the practice.

However, from different points of view, it is not satisfactory neither of the above description levels.Clearly, descriptions at the software level are unsuited for the explanation of the working of the system.For example, according to a software description, the "memory may be thought of as mailboxes containing groups of ones and zeros "./[2]/. Since the exchange of the content of a mailbox is no problem, the mailbox model of the memory cell cannot give account of the problem of the reading and writing in a random access memory.

Descriptions at the hardware level may be sufficient and also necessary by the planning of computers. But it is superfluous to speak about electric circuits of semicondoctor chips, etc. if it is wanted to understand only the function and the role of a component which is in question. Moreover, it is on the point of impossibility to give a well-arranged hardware description in the case of an LSI or VLSI circuit, where the number of transistors is more than 10000.

Thus, it seems to be evident, to give system theoretical models for computers or components of them. Namely, it does not matter to a system theorist whether a system is electrical, mechanical, optical, economic, biological etc. It is of interest to him the mathematical structure and not the physical form of a system /see Zadeh [12]/.

Thus, we have to investigate the possibility of the application of traditional mathematical tools and methods for the description of the behaviour of computers and their components. The idea of application of mathematical means in the abstract describtion of electronic digital computers goes back to L. KALMAR, who, yet in 1962, gave an abstract mathematical model for automatic digital computers of that time /as a tuple of eleven things/ based on the notion of the Mealy automata /[8]/. But, till now, nobody used neither this nor some other definition of computer to develop an abstract mathematical theory.

In the present work, following a new approach of the notion of the memory, we develop a mathematical theory of processors. According to our definition, a memory on a set S is not other than a threetuple $\mathcal{M}=(S,C,s)$, where S is the set of states of the memory $\mathcal{M}$, C — the core of $\mathcal{M}$ — is a family of partitions of S and s — the instant state of $\mathcal{M}$ — is a variable on S.

Every partition $P \in \mathbb{C}$ is called a memory cell of $\mathcal{M}$. Let the classes of P be indexed in some way with a low segment of non-negative integers. Every instant state s is contained in exactly one klass $K_n \in P$: $s \in K_n$. We shall say in this case that the content of the memory cell P is n at the instant state s. Thus, the content of a memory cell depends on the instant state.

Now, if we create a memory on the state set of an automaton, then a processor comes into being which operates on the content of the memory, in accordance with the classical determination of the processor /[2]/.

In Section 1., as a technical preparation, we discuss the question, in which way is possible to describe the partitions. Section 2.deals with the properties of the memory /independence, maximality, capacity/. Finally, in Section 3., we treat the processors and some other connected questions.

I have lectured on some parts of this work in several conferences. I have also used the matherial as a two-semester undergraduate course at L. Kossuth University in Debrecen. I have recived encouragement, criticism and other valuable help from many colleagues and friends, and from my wife. Thanks are due to all of them, but in particular to M. Arató, Z. Daróczy, J. Demetrovics. I am grateful to P. Dömösi for a number of helpful comments. A number of useful criticisms were supplied by Monika Tömösvári, Alberto, A.I. and the others, as students of my.

0. DENOTATIONS

We think that the world consists of things, sets of things and relations between things. In order to avoid some inaccuracies, we deviate to a certain extent from the standard notation.

The set of binary relations on a set S will be denoted by $REL(S)$. The elements of $REL[REL(S)]$ will be called hyper-relations. The equality hyper-relation will be denoted by $\leftrightarrow$ which may be defined as follows

$$(0.1) \qquad \forall \sigma,\tau \in REL(S): \sigma \leftrightarrow \tau \iff \forall x,y \in S \ (x \sigma y \iff x \tau y) .$$

We denote by $EQU(S)$ the set of equvalance relations on S. For all $\chi \in EQU(S)$ the equivalence class of an element s is the set $\chi s = \{x \in S \mid x \chi s \}$.
The cardinality of a set S will be denoted by $|S|$. For nonempty sets S, H, the set of all functions $f: S \to H$ will be denoted by $F(S,H)$.

The set of injective, bijective and surjective functions $f \in F(S,H)$ will be denoted by $INJ(S,H)$, $BIJ(S,H)$ and $SUR(S,H)$, respectively. Thus,

$$INJ(S,H) = \{f \in F(S,H) \mid \forall s_1, s_2 \in S [f(s_1)=f(s_2) \iff s_1=s_2]\},$$

$$SUR(S,H) = \{f \in F(S,H) \mid f(S) = H \} ,$$

$$BIJ(S,H) = INJ(S,H) \cap SUR(S,H)$$

where $f(S)$ is the range of f : $f(S) = \{h \in H \mid h=f(s), s \in S \}$.

1. PARTITIONS

As was pointed out in the introduction, the abstract definition of the memory will be based on the notion of the partition of a set. Before the studying of the memory, we discuss the question:in which way is possible to describe partitions of sets in a suitable manner to our purposes.

We can describe partitions of a set S in a convenient way by means of equivalence relations. If χ is an equvalence relation on the set S then the family

$$(*) \qquad S/\chi := \{\chi s \mid s \in S\}$$

of equivalence classes is a partition of S. Every partition of S may be represented uniquely in this way. Thus, we turn our attention to the description of equivalence relations.

In the sequel, we shall suppose that S and H denote nonempty sets.

DEFINITION 1.1. For all $f \in F(S,H)$ we define a binary relation $(f) \in REL(S)$ in the following manner: $\forall x,y \in S$:

$$(1.1) \qquad x(f)y \iff f(x)=f(y)$$

REMARK 1.1. Clearly, if f is injective, then $(f) \leftrightarrow = $.

PROPOSITION 1.1. (i)

$$(1.2) \qquad \forall f \in F(S,H): \quad (f) \in EQU(S).$$

(ii) If $|S| \leq |H|$, then

$$(1.3) \qquad \forall \chi \in EQU(S) \exists f \in F(S,H): \chi \leftrightarrow (f) .$$

Proof.(i) On the basis of Definition 1.1, it is easy to check that (f) is a reflexive, symmetric and transitve relation.
(ii) Let $\chi \in EQU(S)$ be an arbitrary eqivalence relation. Let S/χ be the $(*)$ classification. Since $|S/\chi| \leq |S| \leq |H|$, there exists an injective map $\varphi \in F(S/\chi,H)$. Let $f(s) = \varphi(\chi s)$ for arbitrary $s \in S$. Then, clearly, $f \in F(S,H)$, moreover, using that φ is injective, we obtain $\forall x,y \in S$:

$$(1.4) \qquad x(f)y \iff f(x)=f(y) \iff \varphi(\chi x)=\varphi(\chi y) \iff \chi x=\chi y \iff x \chi y$$

which shows that $(f) \leftrightarrow \chi$. ■

DEFINITION 1.2. ON the basis of Proposition 1.1, we introduce the following terminology: if $x(f)y$ holds for some $x,y \in S$ then we say that "x and y are identical according to f ".

REMARK 1.2. Using the above terminology, Remark 1.1 and Proposition 1.1 may be formulated as follows:
(a) The identity according to a function is always an equivalence relation which is restricted to the equality relation in case of an injective function f .
(b) Any equivalence relation is an identity according to a suitable function f .

REMARK 1.3. For the relation (f) , some autors use the term "kernel of the mapping f " /see e.g. [6]/. But, in other area of the literatur, the term "kernel of f " is reserved for the set of zeros of f . The name "identity according to f " is borrowed from the theory of colour recoonition where f is a signal transformation /see e.g. [4] /.

It may happen that for different functions f and g, the identity according to f is the same relation as the identity according to g. The next theorem gives a necessary and sufficient condition to this case.

THEOREM 1.1. Let $f:S \to H_1$ and $g:S \to H_2$ be surjective functions. The identity according to f agrees with the identity according to g iff there exists a bijective map $\Psi:H_1 \to H_2$ such that

$$g = \Psi(f) \tag{1.5}$$

Proof. Sufficiency. Suppose that (1.5) is valid for some $f \in SUR(S,H_1)$, $g \in SUR(S,H_2)$ and $\Psi \in BIJ(H_1,H_2)$. Then $\forall x,y \in S$:

$$\Psi[f(x)] = \Psi[f(y)] \iff f(x) = f(y), \tag{1.6}$$

since Ψ is injective. Thus,

$$\forall x,y \in S : \tag{1.7}$$

$$x(g)y \iff g(x)=g(y) \iff \Psi[f(x)]=\Psi[f(y)] \iff f(x)=f(y) \iff x(f)y$$

which shows that

$$(f) \leftrightarrow (g) \quad . \tag{1.8}$$

Necessity. Let (1.8) be true for $f \in SUR(S,H_1)$, $g \in SUR(S,H_2)$. Our task is to construct the function $\Psi \in BIJ(H_1,H_2)$ with property (1.5). For this reason, we pick out an element from every class of the classification $S/(f)$ and we mark these elements by a star. Now, let $u \in H_1$ be given arbitrarily. Since f is surjective, there exists a marked element $s^* \in S$ such that

$$f(s^*) = u \ . \tag{1.9}$$

Then, let

$$\Psi(u) = g(s^*) \tag{1.10}$$

In this way, we have defined a function $\Psi \in F(H_1,H_2)$. We shall show that Ψ is injective. Let $u_1,u_2 \in H_1$ be such that

$$\Psi(u_1) = \Psi(u_2) \ . \tag{1.11}$$

Since f is surjective, there exist marked elements s_1^* and s_2^* such that

(1.12) $f(s_1') = u_1 \ , \quad f(s_2'') = u_2 \ .$

Then, by (1.10) and (1.11), we obtain that

$$g(s_1') = \Psi(u_1) = \Psi(u_2) = g(s_2'')$$

Thus, in consequence of (1.8), by Definition 1.1, we get

$$g(s_1')=g(s_2'') \iff s_1'(g)s_2'' \iff s_1'(f)s_2'' \iff f(s_1')=f(s_2'')$$

from which, by (1.12), it follows $u_1 = u_2$ and so

(1.13) $$\Psi \in INJ(H_1, H_2) \ .$$

Now, we show that Ψ is surjective. Since g is surjective, for arbitrary $v \in H_2$ there exists $s' \in S$ such that $g(s') = v$. Let s^* be the marked element of the class $(f)s'$, i.e. $s^*(f)s'$ from which, by (1.8), it follows also $s^*(g)s'$ i.e.

(1.14) $$g(s^*) = g(s') = v \ .$$

Thus, for $u=f(s^*) \in H_1$, by (1.10), we get $\Psi(u)=g(s^*)=v$ and $\Psi \in SUR(H_1, H_2)$ which together with (1.13) imply $\Psi \in BIJ(H_1, H_2)$.

Finally, we prove (1.5). Let s be an arbitrary element of S and s^* be the marked element of the class $(f)s$. Then

(1.15) $$s^*(f)s \iff f(s^*)=f(s)$$

whence, by (1.8),

(1.16) $$s^*(g)s \iff g(s^*)=g(s) \ .$$

Thus, for the element $u=f(s^*)$ by (1.10), (1.15), (1.16) we have

$$\Psi[f(s)] = \Psi[f(s^*)] = \Psi(u) = g(s^*) = g(s)$$

which shows that (1.5) is true and the theorem is proved. ■

If the function f is not injective, the customary inverse function of f doesn't exist. Therefore, one defines the inverse of f as the inverse relation of f /considering f as a relation/. We shall here define the inverse of f as the set-valued function

(1.17) $$f^{-1}(h) = \{s \in S \mid f(s)=h, \ h \in f(S)\}$$

which exists always too.

Now, we shall show that the equivalence classes of the classification $S/(f)$ may be obtained as the sets $f^{-1}(h)$.

THEOREM 1.2. If f^{-1} is the inverse of the function $f: S \to H$ then

(1.18) $$f^{-1}: f(S) \to S/(f) \ ,$$

and

(1.19) $$f^{-1} \in BIJ[f(S), S/(f)] .$$

Proof. Let $f \in F(S, H)$. In order to prove (1.18), we consider an arbitrary element $h \in f(S)$. So, $f^{-1}(h) \neq \emptyset$ /not empty/. Let $s_0 \in f^{-1}(h)$ be a fixed element and $s' \in f^{-1}(h)$ be an arbitrary element.

Then, we have $f(s')=h=f(s_0)$ i.e. $s' \in (f)s_0$ which shows that

(1.20) $$f^{-1}(h) \subseteq (f)s_0 \ .$$

On the other hand, if $s \in (f)s_0$ then $s \sim (f)s_0$ i.e. $f(s)=f(s_0)=h$ showing that $s \in f^{-1}(h)$. Thus, $(f)s_0 \subseteq f^{-1}(h)$ which with (1.20) yield $f^{-1}(h)=(f)s_0$ and (1.18) is proved.

In order to prove (1.19), first we show that f^{-1} is surjective. Let $(f)s_1$ be an arbitrary class of $S/(f)$. For the element $h_1=f(s_1)$, obviously, $h_1 \in f(S)$, moreover

$$f^{-1}(h_1)=\{s \in S \mid f(s)=h_1\}=\{s \in S \mid f(s)=f(s_1)\}=\{s \in S \mid s \sim (f)s_1\}=(f)s_1$$

and thus, f^{-1} is surjective.

We show that f^{-1} is injective. Let $h_1, h_2 \in f(S)$ be such that $f^{-1}(h_1) = f^{-1}(h_2)$. Then $s \in f^{-1}(h_1)=f^{-1}(h_2)$ implies $h_1=f(s)=h_2$ and so f^{-1} is injective. Since f^{-1} is both injective and surjective, it is bijective. ■

COROLLARY 1.1. Consequences of Theorem 1.2: $\forall f \in F(S,H)$

(1.21) $$|f(S)| = |S/(f)| \ ,$$

(1.22) $$f^{-1}[f(S)] = S/(f) \ .$$

DEFINITION 1.4. Let P be a finite partition of the set S. A function $f \in SUR(S,N)$ is called a base function for P if the following properties are fulfilled:

(a) N is a low segment of numbers:

(1.23) $$N = \{0,1,\ldots,|P|-1\}$$

(b) P may be written in the form

(1.24) $$P = f^{-1}(N) \ .$$

THEOREM 1.3. Let P be a finite partition

(1.25) $$P = S/\chi \qquad /\chi \in EQU(S)/$$

and let

(1.26) $$f(s) = \varphi(\chi s) \qquad / s \in S /$$

where

(1.27) $$\varphi \in BIJ(P,N) \qquad /N=\{0,1,\ldots,|P|-1\}/.$$

Then f is a base function for P. Conversely, any base function for P may be written in the form (1.26) by means of a φ (1.27).

Proof. Let P be the partition (1.25) and let f be composed in the form (1.26) by means of a function φ with property (1.27). Then, clearly,

(1.28) $$f(S) = \varphi(P) = N \ ,$$

moreover,

(1.29) $$(f) \leftrightarrow \chi \ .$$

/see (1.4)/. Whence, by (1.22)

$$f^{-1}(N) = f^{-1}[f(S)] = S/(f) = S/\chi = P$$

which shows that f is a base function of P.

Conversely, let $f: \mathbb{S} \to \mathbb{N}$ be a base function for the partition (1.25). Then, by Definition 1.4, $P = f^{-1}(\mathbb{N})$ and f is surjective. Let the funtion $\varphi: P \to \mathbb{N}$ be defined as follows

$$\varphi[f^{-1}(i)] = i \qquad / \forall i \in \mathbb{N} /. \tag{1.30}$$

For arbitrary $s \in \mathbb{S}$, there exists uniquely an $i \in \mathbb{N}: s \in f^{-1}(i)$, i.e.

$$f(s) = i \tag{1.31}$$

and thus,

$$f^{-1}(i) = \chi s \ . \tag{1.32}$$

It follows from (1.21), (1.30) and (1.32)

$$f(s) = i = \varphi[f^{-1}(i)] = \varphi(\chi s) \ .$$

This shows that f is of the form (1.26) and it is clear by (1.30) that also (1.27) is fulfilled. ■

DEFINITION 1.5. Let P be a partition of the set $\mathbb{S}$. The function $g \in SUR(\mathbb{S}, \mathbb{H})$ is said to be a generator function for P if

$$g^{-1}(\mathbb{H}) = P \ . \tag{1.33}$$

COROLLARY 1.2. Let f be a fixed base function for the finite partition P. Then any generator function g for P may be written in the form $g = \Psi(f)$ with an appropriate $\Psi \in BIJ(\mathbb{N}, \mathbb{H})$.

DEFINITION 1.6. A partition P of $\mathbb{S}$ is said to be proper if

$$|P| > 1$$

or, equivalently, if the generator function for P is not constant.

2. MEMORY

A software description of the memory is the following/[2]/: "Memory is the device where information is stored. ... One may view information as being stored in the form of ones and zeros. ... Memory may be thought of as mail boxes containing groups of ones and zeros."

The hardware description of the memory is not so simple. There are known several types of the memory depending on the hardware realizations. /See [10]/.

Before the discussion of the abstract mathematical notion of the memory, I should like illustrate the essence of the memory from practical side. Almost anything may serve as a device to store information provided the following conditions are fulfilled / every veritable thing is in some state at any instant and the undermentioned conditions refer to the instant states of the device/: (a) The number of the possible different states of the device is at least two. (b) The instant state of the device may be altered at will. (c) The state remains essentially unchanged till it will be not altered.

So, for example, a handkerchief may be used as a memory.The number of the possible instant states of a handkerchief is infinite. A handkerchief may be ironed, it may be creased with an infinite number of the possible wrinkles. In a handkerchief may be tied knots. At the same time the handkerchief may be clean, fragrant or it can be dirty and stinking, etc. In general, an instant state of the handkerchief may be characterized by a vector.

In order to use the handkerchief as a device where information will be stored, it is necessary to make a number of knots in the handkerchief. Thus, from the point of view of the storage of information we don't make distinction between the states s_1 ,s_2 of the handkerchief if the number of the tied knots is the same in both states s_1 , s_2 . In this way, we created a partition in the set of states of the handkerchief. The equivalence classes are the followings: the set of states of zero knot, the set of states of one knot and so on until a finite number K of knots.

Naturally, it is needed to clear, what do the cases of zero, one,...,K knots mean.

It seems also from this simple example that the storage of information in a device goes together with the creation of a partition in the set of states of the device. Moreover, this example shows that information may be stored not only "in the form of ones and zeros".

The software way of thinking in "ones and zeros" goes back to the hardware realization of the memory. For example, the storage of information is based on capacitors in the dynamic RAM chips. The state of a capacitor is determined by its voltage. Thus, the set of states of a capacitor is infinite. It is artificially established a partition in the set of states: the class of low voltage states and the class of high voltage states. If the instant state of the capacitor is contained in the first or second class then one says that the capacitor stores the zero or one, respectively. The transistors are used here only to alter the instant state /write/ and to the perception of the instant state /read/ of the capacitor.

Moreover, it is needed to take care of the fulfilment of condition (c) since the charge of the capacitor will be lost /refresh circuits, see [10]/.

2.1. Finite memory and memory on a finite set

DEFINITION 2.1. Let $\mathbb{F}$ be a set of functions defined on $\mathbb{S}$. We say that $\mathbb{F}$ consists of strong different functions if

$$\forall f,g\in\mathbb{F} : (f) \leftrightarrow (g) \Rightarrow f = g \quad .$$

REMARK 2.1. Clearly, different elements of a set of strong different functions determine different partitions of $\mathbb{S}$.

DEFINITION 2.2. Let $\mathbb{C}$ be a set of partitions of a non-void set $\mathbb{S}$ and let s be a variable on $\mathbb{S}$. We consider the threetuple

(2.1) $$\mathcal{M} = (\mathbb{S},\mathbb{C},s)$$

and introduce the following names:

(i) $\mathcal{M}$ will be called a memory on the state set $\mathbb{S}$. The set $\mathbb{C}$ is called the core of $\mathcal{M}$. The value of the variable s will be called the instant state of the memory $\mathcal{M}$.

(ii) Any partition $P \in \mathbb{C}$ is said to be a memory cell of $\mathcal{M}$.

(iii) Let $\mathbb{F}$ be a set of strong different functions defined on $\mathbb{S}$. The core $\mathbb{C} = \{P=f^{-1}[f(\mathbb{S})] | f\in\mathbb{F}\}$ will be denoted by $\langle\mathbb{F}\rangle$ and $\mathbb{F}$ is called the generator set of the core $\mathbb{C}$.

(iv) Since $P = f^{-1}[f(\mathbb{S})]$ is a partition of $\mathbb{S}$, the instant state s is contained in exactly one P class: $s\in f^{-1}(h)$ /$h\in\mathbb{H}=f(\mathbb{S})$, $f\in\mathbb{F}$/. We say in this case that the content of the memory cell P is h at the instant state s. Denoting the content of P at the instant state s by cont(P)|s, we can write

(2.3) $$\mathrm{cont}(P)|s = h \iff s\in f^{-1}(h)$$

If the content of P is h at the instant state s then, as usual, we say that h is stored in the memory cell P at the state s. Since $s\in f^{-1}(h)$ is equivalent to $h=f(s)$, we may write

(2.4) $$\mathrm{cont}\{f^{-1}[f(\mathbb{S})]\}|s = f(s)$$

instead (2.3).

(v) The capacity of memory cells is defined for finite cells. A memory cell P is finite if P cosists of a finite number of equivalence classes /which need not all be finite subsets of $\mathbb{S}$/.

(2.5) $$\mathrm{cap}(P) = \log_2 |P| \qquad /P\in\mathbb{C}/.$$

where cap(P) denotes the capacity of the memory cell P in bits.

DEFINITION 2.3. Let $\mathcal{M}$ be a memory over a set $\mathbb{S}$. If the core of $\mathcal{M}$ consits of a finite number of finite memory cells then $\mathcal{M}$ is said to be finite.

Clearly, any memory, defined over a finite set $\mathbb{S}$, is finite. The capacitor memory serves as an important example for a finite memory which is defined over an infinite set. Its mathematical description will be given in the following example.

EXAMPLE 2.1. 1 Bit Dynamic Memory. The instant state of a capacitor is determined by its voltage which may be an element of the whole set of real numbers. But here, one uses only positive voltage. Moreover, in order to put in practice the safe distinction of the two essentially different classes of instant states, it is established an interval $[L,H]$ of voltages which is not used. Thus,

$$S = (0,L) \cup (H,\infty)$$

where $0 < L < H$. /Theoretically, it would be sufficient to take the case when $L = H$ /.

The set F of strong different funtions at present consists of the single base function

$$f(s) = \begin{cases} 0 & \text{if } 0 < s < L \\ 1 & \text{if } H < s \end{cases}$$

Thus, $f(S) = N = \{0,1\}$, $f^{-1}(0) = (0,L)$, $f^{-1}(1) = (H,\infty)$ and $f^{-1}(N) = \{(0,L),(H,\infty)\} = S/(f) = P$. The capacity of the memory cell P is $cap(P) = \log_2 |P| = \log_2 2 = 1$ bit .
Thus, the mathematical model of the 1 bit dynamic memory is

(2.7) $$M = [(0,L) \cup (H,\infty), \langle\{f\}\rangle, s]$$

where $L = 0.4$ V , $H = 2.4$ V by a typical case in practice, moreover the voltage is not to be more than 5 V so, we may write $(H,5]$ instead (H,∞) .

REMARK 2.2. Naturally, the capacity of the memory cell P refers to the storage of information and $cap(P)$ is independent on the electric capacitance of the capacitor. In the practice, the access time, the refresh cycle and other parameters are depending on the capacitance of the capacitors in dynamic RAM chips.

REMARK 2.3. The hardware description of a dynamic memory is more complicated as the mathematical description given here. The reason for this is twofold: 1/. The hardware realization of a partition in the state space is not so simple as the mathematical construction by means of a base function. 2/. By any hardware realization, it is solved also the problem of the reading and writing. So, the hardware description is yet a description of a RAM which is a more complicated device.
We shall deal later with the mathematical description of RAM .

REMARK 2.4. The formula (2.7) will serve as an abstract characterization not only of a dynamic memory cell made of a capacitor /and other electric instruments/ but also of its mechanical equivalent where a vessel filled with water plays the role of the charged capacitor. The instant state will be determined by the water-level in the vessel.

Naturally, in practice, a memory based on vessels doesn't exist, it would be excessively large and very slow.

DEFINITION 2.4. If the core C of a memory M is a one element set then M is said to be unicellular. If C contains more than one memory cells then M is said to be a multicellular memory.

2.2. The independence of the memory

Next, we raise the question about the independence of the memory cells. A memory cell P is independent of the others in $\mathcal{M}$ if it is possible to change the content of P without disturbing the content of other memory cells in $\mathcal{M}$. More precisely:

DEFINITION 2.5. Let $\mathcal{M}$ be a memory on a set S and P be a memory cell of $\mathcal{M}$. P is said to be independent in $\mathcal{M}$ if the following conditions are fulfilled: (i) It is possible to alter the content of P. (ii) It is possible carry through any alteration of the content of P in such a way that the contents of the other cells of $\mathcal{M}$ remain unchanged.
If every memory cell of $\mathcal{M}$ is independent in $\mathcal{M}$ then we say that $\mathcal{M}$ is independent.

THEOREM 2.1. A unicellular memory on a set S is independent iff its memory cell is a proper partition of S.

Proof. If the partition P is not proper then $P = \{S\}$ and so, it is impossible to change the content of P. Consequently, by Definition 2.5, P can't be independent in no memory.
On the other hand, if the partition is proper then both conditions (i) and (ii) of Definiton 2.5 are fulfilled trivially and so, P is independent in the memory $\mathcal{M}$ which is in question. Since $\mathcal{M}$ has no more memory cells, also $\mathcal{M}$ is independent. ■

REMARK 2.5. It's easy to construct a memory $\mathcal{M}$ which is not independent in spite of the fact that every memory cell of $\mathcal{M}$ is a proper partition. For example, let $\mathcal{M}$ be a multicellular memory of which memory cells are made of wessels mentioned in Remark 2.4. Obviously, if $\mathcal{M}$ has memory cells with communicating vessels then $\mathcal{M}$ can't be independent. Namely, neither of the communicating vessels can determine an independent memory cell in $\mathcal{M}$ since one of these cells will have always the same content as the other.

PROPOSITION 2.1. Let $\mathcal{M} = (S, \langle F \rangle, s)$ be a multicellular memory. The memory cell $P = f^{-1}[f(S)] \quad /f \in F/$ is independent in $\mathcal{M}$ iff P is a proper partition of S and

$$(2.9) \qquad \forall s, s' \in S : f^{-1}[f(s')] \cap \bigcap_{g \in F \setminus \{f\}} g^{-1}[g(s)] \neq 0$$

Proof. Necessity. Suppose that P is independent in $\mathcal{M}$. Then, P is proper, as we saw by the proof of Theorem 2.1. In order to prove (2.9), we show that the negation of (2.9) leads to the contradiction that P is not independent. The negation of (2.9)

$$\exists s, s' \in S : f^{-1}[f(s')] \cap \bigcap_{g \in F \setminus \{f\}} g^{-1}[g(s)] = 0 .$$

This is equivalent to

$$(2.10) \qquad \exists s, s' \in S : \forall s'' \in f^{-1}[f(s')] : s'' \notin \bigcap_{g \in F \setminus \{f\}} g^{-1}[g(s)].$$

Since

$$s'' \notin \bigcap_{g \in \mathbf{F} \setminus \{f\}} g^{-1}[g(s)] \iff \exists\, g \in \mathbf{F} \setminus \{f\}: s'' \notin g^{-1}[g(s)] ,$$

moreover

$$s'' \in f^{-1}[f(s')] \iff f(s'') = f(s')$$

and

$$s'' \notin g^{-1}[g(s)] \iff g(s'') \neq g(s) ,$$

we may formulate (2.10) as follows: There exist $s,s' \in \mathbf{S}$ so that for every instant state $s'' \in \mathbf{S}$ in which $cont(P)|s''=f(s'')=f(s')=h'$ there exists a $g \in \mathbf{F}$ $/g \neq f/$ generating a $Q = g^{-1}[g(\mathbf{S})] \neq P$ memory cell such that $cont(Q)|s'' = g(s'') \neq g(s) = cont(Q)|s$.
Thus, it is impossible to alter the content of P from $cont(P)|s= f(s) = h$ to $cont(P)|s' = h'$ without any alteration of the content of some $Q \neq P$ memory cell. Therefore, P is not independent in $\mathcal{M}$ which is a contradiction and the necessity of the condition is proved.
Sufficiency. Let $f \in \mathbf{F}$ be a function such that the partition $P = f^{-1}[f(\mathbf{S})]$ is proper and (2.9) holds. Since

$$s'' \in \bigcap_{g \in \mathbf{F} \setminus \{f\}} g^{-1}[g(s)] \iff \forall\, g \in \mathbf{F} \setminus \{f\}: s'' \in g^{-1}[g(s)] \iff$$

$$\iff \forall\, g \in \mathbf{F} \setminus \{f\}: g(s'') = g(s) ,$$

the formula (2.9) may be read as follows:
Let s be an arbitrary instant state of the memory and $h'=f(s')$ be arbitrarily chosen as a new content of P. Then there exists an instant state s'' such that

$$cont(P)|s'' = f(s'') = f(s') = h'$$

whilst for every memory cell $Q = g^{-1}[g(\mathbf{S})]$ of $\mathcal{M}$ if $Q \neq P$ then

$$cont(Q)|s'' = g(s'') = g(s) = cont(Q)|s ,$$

i.e. it is possible to perform any change of the content of P such that the content of the other memory cells in the core of $\mathcal{M}$ remains unchanged. Thus, P is independent in $\mathcal{M}$ and we showed that the condition (2.9) is sufficient for a proper P to be independent in $\mathcal{M}$. ■

COROLLARY 2.1. Let $\mathcal{M}=(\mathbf{S},\langle\mathbf{F}\rangle,s)$ be a multicellular memory such that every memory cell of $\mathcal{M}$ is a proper partition of $\mathbf{S}$. Then a necessary and sufficient condition for $\mathcal{M}$ to be independent is

$$(2.11) \quad \forall f \in \mathbf{F}\ \forall\, s,s' \in \mathbf{S}: f^{-1}[f(s')] \cap \bigcap_{g \in \mathbf{F} \setminus \{f\}} g^{-1}[g(s)] \neq \emptyset .$$

DEFINITION 2.6. Let $\mathbf{F}$ be a finite set of strong different functions defined on a set $\mathbf{S}$. Let $(f_1,\ldots,f_n)$ be an array of the elements of $\mathbf{F}$. We define a function $F_0: \mathbf{S} \to H_1 \times \ldots \times H_n$, where $H_i = f_i(\mathbf{S})$, $i \in \Gamma_n=\{1,\ldots,n\}$, such that $\forall\, s \in \mathbf{S}$

$$(2.12) \qquad F_0(s) = [f_1(s),\ldots,f_n(s)] .$$

The unicellular memory $\mathcal{M}_0 = (\mathbf{S},\langle\{F_0\}\rangle,s)$ will be called the vector form of the memory $\mathcal{M} = (\mathbf{S},\langle\mathbf{F}\rangle,s)$. The memory cell of $\mathcal{M}_0$ will be called the vector cell of the memory $\mathcal{M}$.

REMARK 2.6. It is evident from the above definition that every unicellular memory agrees with its vector form. If $\mathcal{M}$ is a multicellular memory then one can store in the memory cell $P_1=f_1^{-1}(H_1)$ an element $h_1 \in H_1$ and so on, in the memory cell $P_n=f_n^{-1}(H_n)$ an element $h_n \in H_n$. The vector cell of $\mathcal{M}$ /which is generated by the function (2.12)/ is suited for the storage of vectors

$$h = (h_1,\ldots,h_n) \in H_1 \times \ldots \times H_n \quad . \tag{2.13}$$

Clearly, a necessary and sufficient condition for the possibility to store an arbitrary vector (2.13) in the vector cell of $\mathcal{M}$, is the surjectivity of the function (2.12). It is evident that the surjectivity of F_0 is independent of the order of the elements $f_1,\ldots,f_n$.

The next theorem shows that the surjectivity of F_0 is equivalent to the independence of the memory $\mathcal{M}$ / not of $\mathcal{M}_0$. Namely, the vector form of $\mathcal{M}$ is a unicellular memory, which may be independent so that $\mathcal{M}$ is not independent /.

THEOREM 2.2. Let $\mathcal{M}$ be a multicellular finite memory with proper memory cells. $\mathcal{M}$ is independent if and only if the generator function of the vector cell of $\mathcal{M}$ is surjective.

Proof. Sufficiency. Let the vector function (2.12) be surjective. We shall show that every memory cell $P_i=f_i^{-1}[f_i(S)]$ /$i\in\{1,\ldots,n\}$/ is independent in $\mathcal{M}$. Since P_i is proper, it remains to prove that any alteration of the content of P_i may be carried through such that the content of the other memory cells P_j / $j \neq i$ / remain unchanged. Formally, we have to prove the validity of the formula

$$\forall i\in\Gamma_n \forall s,s' \in S\ \exists s'' \in S$$
$$\mathrm{cont}(P_i)|s''=\mathrm{cont}(P_i)|s' \ :\Rightarrow\ \forall j\in\Gamma_n\setminus\{i\}:\mathrm{cont}(P_j)|s''=\mathrm{cont}(P_j)|s \quad .$$

For $s,s' \in S$, let $h = [f_1(s),\ldots,f_n(s)]$, $h' = [f_1(s'),\ldots,f_n(s')]$ and $h'' = [f_1(s),\ldots,f_{i-1}(s),f_i(s'),f_{i+1}(s),\ldots,f_n(s)]$.
Since $F_0 \in \mathrm{SUR}[S, H_1\times\ldots\times H_n]$, there exists an $s'' \in S$ such that $F_0(s'') = h''$. Then $f_1(s) = f_1(s''),\ldots,f_{i-1}(s) = f_{i-1}(s'')$, $f_i(s')=f_i(s'')$, $f_{i+1}(s)=f_{i+1}(s''),\ldots,f_n(s)=f_n(s'')$ and so $\mathrm{cont}(P_i)|s' = f_i(s') = f_i(s'') = \mathrm{cont}(P_i)|s''$ and for $j \in \{1,\ldots,n\}\setminus\{i\}$ $\mathrm{cont}(P_j)|s = f_j(s) = f_j(s'') = \mathrm{cont}(P_j)|s''$. Thus, we have proved that $\mathcal{M}$ is independent.
Necessity. Supposing the independence of $\mathcal{M}$, we shall prove that F_0 is surjective. Let $h = (h_1,\ldots,h_n) \in H_1\times\ldots\times H_n$ be given arbitrarily. In order to prove that $\exists\, s \in S: F_0(s) = h$, it is sufficient to show that there exists a procedure for the solution of the equation

$$F_0(x) = h \quad . \tag{2.14}$$

Let $s_0 \in S$ be an arbitrary starting element to the procedure which will be given below in BASIC language. In order to make easier the reading of the program, we want not change the notation used till now. Thus, we remain by the usual index denotation and don't use the BASIC formality of arrays.

/One may recommend other high level programming languages which can be more suitable for description of algorithms. But, I think that the BASIC is the most popular language which is applicable conveniently to the present simple case/.

```
1 LET s=s0
2 FOR i=1 TO n
3 IF fi(s)≠hi THEN GOTO 7
4 NEXT i
5 LET x=s
6 STOP
7 REM Since hi ∈ Hi = fi(S), there exists an s' ∈ S
  such that (1)    fi(s') = hi . Thus, in any concrete
  case, it is possible to find a program which solves
  the equation (1). In order to write the present
  subroutine, one must make a such program in any
  concrete case and write it instead of this REM
  statement.
8 REM Since M is independent, any memory cell of M
  is independent in M. Thus, for s,s'∈S ∃ s"∈ S :
  (2)        fi(s") = fi(s')
  (3)     ∀j∈{1,...,n}\{i}: fj(s") = fj(s) .
  On the basis of this fact, one can write a program
  in order to obtain an s" as solution of (2), (3).
  We must write this program instead of this  REM
  statement.
  9 LET s=s"
10 GO TO 2
```

It is clear that the result of this program satisfies the equation (2.14) in any concrete case and the surjectivity of F_0 is proved.■

COROLLARY 2.2. Let $M=(S,(\{f_1,\ldots,f_n\}),s)$ be a memory with non-constant generator functions. M is independent if and only if

(2.15) $\forall h \in H_1 \times \ldots \times H_n : f_1^{-1}(h_1) \cap \ldots \cap f_n^{-1}(h_n) \neq 0$ /$h=(h_1,\ldots,h_n)$/.

Proof.We show that (2.15) is equivalent to the surjectivity of the generator function of the vector cell of M .
$\forall h \in H_1 \times \ldots \times H_n : f_1^{-1}(h_1) \cap \ldots \cap f_n^{-1}(h_n) \neq 0 \iff \forall h \in H_1 \times \ldots \times H_n \exists s \in S : s \in f_1^{-1}(h_1) \cap \ldots \cap f_n^{-1}(h_n) \iff \forall h \in H_1 \times \ldots \times H_n \exists s \in S : s \in f_1^{-1}(h_1) : \ldots : s \in f_n^{-1}(h_n) \iff \forall h \in H_1 \times \ldots \times H_n \exists s \in S : h_1 = f_1(s) : \ldots : h_n = f_n(s) \iff \forall h \in H_1 \times \ldots \times H_n \exists s \in S : h = F_0(s) \iff F_0 \in SUR(S, H_1 \times \ldots \times H_n)$.
Thus, by Theorem 2.2, the independence of M is equivalent to the validity of the formula (2.15). ■

2.3. The capacity of the memory

Until now, we have defined the capacity of a memory cell but not of the whole memory. Now, subsequently, we give the more general definition and we discuss some fundamental properties of the capacity.

DEFINITION 2.7. By the capacity of a finite memory $\mathcal{M}=(S,C,s)$, we mean the capacity of the core of $\mathcal{M}$: $cap(\mathcal{M}) := cap(C)$. By the capacity of the core $C = \{P_1,\dots,P_n\}$, we mean the capacity of the vector cell of $\mathcal{M}$:

(2.16) $$cap\{P_1,\dots,P_n\} := cap(P_0)$$

where $P_0 = S/(F_0)$, $F_0(s) = \langle f_1(s),\dots,f_n(s)\rangle$ and $f_1,\dots,f_n$ are generator functions of $P_1,\dots,P_n$, respectively.

REMARK 2.7. The above definition is correct in the sense that $cap(\mathcal{M})$ does not depend on the choice of $F_0 = (f_1,\dots,f_n)$. Indeed, $cap(P_0)=\log_2|F_0(S)|$ is insensible to the permutations of the elements $f_1,\dots,f_n$ of the generator set F .

Every unicellular memory $\mathcal{M}$ agrees with its vector form. Thus, by Definition 2.7, the capacity of $\mathcal{M}$ is equal to the capacity of its memory cell. One may have the guess that the capacity of a multicellular memory $\mathcal{M}$ is equal to the sum of the capacities of the memory cells of $\mathcal{M}$. This is true, but only in case of independence, as the following theorem that shows.

THEOREM 2.3. Let $\mathcal{M} = \langle S,\{P_1,\dots,P_n\}\rangle$ be a finite memory. Then

(2.17) $$cap\{P_1,\dots,P_n\} \le cap(P_1)+\dots+cap(P_n)\ ,$$

where the equality relation holds exactly in case of an independent memory $\mathcal{M}$.

Proof. Let the memory $\mathcal{M}=(S,\langle\{f_1,\dots,f_n\}\rangle,s)$ be independent. Then, by Theorem 2.2, $F_0 \in SUR(S,H_1X\dots X H_n)$, where $F_0 = (f_1,\dots,f_n)$ is the vector cell of $\mathcal{M}$ and $H_i = f_i(S)$, $i\in\Gamma_n$. By Definition 2.7, Definition 2.2.(v), Corollary 1.1,

$$cap(\mathcal{M}) = cap(P_0) = \log_2|P_0| = \log_2|S/(F_0)| = \log_2|F_0(S)| =$$

$$= \log_2|H_1X\dots X H_n| = \log_2[|H_1|.\dots.|H_n|] =$$

$$=\log_2|H_1|+\dots+\log_2|H_n| = \log_2|f_1(S)+\dots+\log_2|f_n(S)| =$$

$$=\log_2|S/(f_1)|+\dots+\log_2|S/(f_n)| = \log_2|P_1|+\dots+\log_2|P_n| =$$

$$= cap(P_1)+\dots+cap(P_n)\ ,$$

i.e. (2.17) is valid by the equality relation. If $\mathcal{M}$ is not independent then, by theorem 2.2,
$|F_0(S)| < |H_1X\dots X H_n|$. Consequently,
$cap(\mathcal{M})=\log_2|F_0(S)| < \log_2|H_1X\dots XH_n|=cap(P_1)+\dots+cap(P_n)$.■

THEOREM 2.4. Let $\mathcal{M}$ be a memory on a finite set S. Then

$$\text{cap}(\mathcal{M}) \leq \log_2 |S| . \tag{2.18}$$

Proof. Let P_0 be the vector cell of $\mathcal{M}$ and F_0 be the generator function of P_0. Then, using Definition 2.7, Definition 2.2.(v) and Corollary 1.1, we get

$$\text{cap}(\mathcal{M}) = \text{cap}(P_0) = \log_2 |P_0| = \log_2 |F_0(S)| . \tag{2.19}$$

On the other hand, it follows from the surjectivity of the function $F_0 : S \to F_0(S)$ that

$$|F_0(S)| \leq |S| . \tag{2.20}$$

The equality (2.19) and the inequality (2.20) yield (2.18).■

REMARK 2.8. The inequality (2.18) is the best estimate for the capacity of memories defined on S. Namely, the equality holds in (2.18) for the unicellular memory $\mathcal{M}$ of which cell P is generated by an injective map $f:S \to H$. In this case $(f) \leftrightarrow = $ and the partition generated by f is the finest partition of S: $P = S/(f)=S/= \; = \{\{s\} | s \in S\}$. Thus, $\text{cap}(\mathcal{M}) = \log_2 |P| = \log_2 |S|$.

Appying these considerations to the generator function of the vector cell of $\mathcal{M}$, we get the following criterion.

PROPOSITION 2.2. The capacity of a memory $\mathcal{M}$ defined on a finite set is the largest possible iff the generator function of the vector cell of $\mathcal{M}$ is injective.

REMARK 2.9. If we suppose that the finite state memory $\mathcal{M}$ is independent then

$$F_0 \in \text{BIJ}(S, H_1 X \ldots X H_n) \tag{2.21}$$

is a necessary condition for

$$\text{cap}(\mathcal{M}) = \log_2 |S| . \tag{2.22}$$

Indeed, if $\mathcal{M}$ is independent then, by Theorem 2.2, F_0 is surjective, moreover, in consequence of (2.22) by Proposition 2.2, it is injective and thus, F_0 is bijective.

Conversely, the following statement is valid: The independence and the maximality of the capacity of $\mathcal{M}$ result from the bijectivity of F_0. Namely, it follows from (2.21) first at all that $|S| = |H_1 X \ldots X H_n| = |H_1| . \ldots . |H_n|$ and that $\mathcal{M}$ is independent because of the surjectivity of F_0. Consequently, (2.22) is true, since, by Theorem 2.3, Definition 2.2.(v),

$$\text{cap}(\mathcal{M}) = \text{cap}(P_1)+\ldots+\text{cap}(P_n) = \log_2 |H_1|+\ldots+\log_2 |H_n| =$$

$$= \log_2 (|H_1| . \ldots . |H_n|) = \log_2 |S| .$$

2.4. Maximal independent memory

Let $\mathcal{M}$, $\mathcal{M}'$ be memories both defined on the same set S. If every memory cell of $\mathcal{M}$ is a memory cell also of $\mathcal{M}'$ then we say that $\mathcal{M}'$ is an extension of $\mathcal{M}$. This relation is an ordering in the set of memories. Every maximal element with respect to this ordering in the set of independent memories on S will be called a maximal independent memory. Clearly, the notion of maximal independent memory differs from the notion of maximal capacity of a memory. But, as we shall see straightway in this section, these notions are not quite independent.

In order to simplify the talks, first, we want get rid of the superfluous improper memory cell and we shall deal with the so-called pure memories.

DEFINITION 2.8. A memory $\mathcal{M}$ on a set S is said to be pure if all memory cells of $\mathcal{M}$ are proper partitions of S. We denote the set of all pure memories on S by MEM(S).

DEFINITION 2.9. We define the union and intersection of memories as follows:

(2.23) $$(S,C_1,s) \cup (S,C_2,s) = (S,C_1 \cup C_2,s),$$

(2.24) $$(S,C_1,s) \cap (S,C_2,s) = (S,C_1 \cap C_2,s).$$

If $C_1 \subset C_2$ then we will write $(S,C_1,s) \subset (S,C_2,s)$ and we shall say that the memory (S,C_2,s) is an extension of the memory (S,C_1,s).

REMARK 2.10. It is possible to work up the present theory in such a way that we require the improper partition $\{\{S\}\}$ to be contained in any core. We may regard $(S,\{\{S\}\},s)$ as the empty memory since $cap(\{\{S\}\}) = \log_2 |\{\{S\}\}| = \log_2 1 = 0$ and it is impossible store information in this cell. In this case the intersection of memories /defined on the same set/ exists always and it is equal at least to the empty memory.

But, for the sake of simplicity, we deal with pure memories. Thus, at the present case, the intersection (2.24) does not exist if $C_1 \cap C_2 = \emptyset$.

DEFINITION 2.10. An independent memory on a set S is said to be maximal if there is no independent extension of $\mathcal{M}$ in MEM(S).

LEMMA 2.1. An independent memory on S is maximal if and only if for every unicellular memory $\mathcal{M}_1 \in$ MEM(S), the memory $\mathcal{M}' = \mathcal{M}_1 \cup \mathcal{M}$ is not independent except when $\mathcal{M}' = \mathcal{M}$.

Proof. Necessity. If $\mathcal{M}$ is maximal then $\mathcal{M}'=\mathcal{M}_1 \cup \mathcal{M}$ can't be independent if $\mathcal{M}' \supset \mathcal{M}$ since in the contrary case $\mathcal{M}'$ would be an independent extension of $\mathcal{M}$ which is impossible by the maximality of $\mathcal{M}$. Sufficiency. Suppose that

(2.25) $$\forall\, \mathcal{M}_1 \in UMEM(\mathcal{S})[\mathcal{M}_1 \cup \mathcal{M} \in IMEM(\mathcal{S}) \Rightarrow \mathcal{M}_1 \cup \mathcal{M}=\mathcal{M}]$$

where $IMEM(\mathcal{S})$ denotes the set of all independent memories on $\mathcal{S}$ and $UMEM(\mathcal{S})$ is the set of all unicellular pure memories on $\mathcal{S}$. Then $\mathcal{M}$ is maximal. We show that the contrary case is impossible. If $\mathcal{M}$ is not maximal then there exixts an independent extension $\mathcal{M}'' \supset \mathcal{M}$. So, the core of $\mathcal{M}''$ contains a partition P'' which is not in the core of $\mathcal{M}$. Thus, $(\mathcal{S},\{P''\},s) \cup \mathcal{M}$ is an extension of $\mathcal{M}$ and so, by (2.25), $\mathcal{M}_1 \cup \mathcal{M}$ can't be independent for $\mathcal{M}_1=(\mathcal{S},\{P''\},s)$. On the other hand, P'' is independent of the other cells in $\mathcal{M}_1 \cup \mathcal{M}$ since $\mathcal{M}_1 \cup \mathcal{M} \subset \mathcal{M}''$ and $\mathcal{M}''$ is independent. Similarly, every memory cell of $\mathcal{M}_1 \cup \mathcal{M}$ is independent in $\mathcal{M}_1 \cup \mathcal{M}$. So, by Definition 2.5, $\mathcal{M}_1 \cup \mathcal{M}$ is independent which is a contradiction. Thus, we have proved that the condition (2.25) is sufficient for the maximality. ■

THEOREM 2.5. Let $\mathcal{M}$ be an independent memory on the finite set $\mathcal{S}$. If

(2.26) $$\log_2 |\mathcal{S}| - 1 < cap(\mathcal{M})$$

then $\mathcal{M}$ is maximal.

Proof. Let $\mathcal{M}$ be an independent memory on $\mathcal{S}$ for which the inequality (2.26) holds. Let $\mathcal{M}_0 \in UMEM(\mathcal{S})$ be such that $\mathcal{M}' = \mathcal{M}_0 \cup \mathcal{M}$ is independent. Let $\{P_0\}$ be the core of $\mathcal{M}_0$ and $\{P_1,\ldots,P_n\}$ be the core of $\mathcal{M}$. In order to prove the maximality of $\mathcal{M}$, on the basis of Lemma 2.1, it suffices to show that $P_0 \in \{P_1,\ldots,P_n\}$. We prove that

(2.27) $$P_0 \notin \{P_1,\ldots,P_n\}$$

leads to a contradiction. Indeed, in this case, the core of $\mathcal{M}'$ is $\{P_0,P_2,\ldots,P_n\}$ and thus, by Theorem 2.3, Definitions 2.7 and 2.2,

$$cap(\mathcal{M}')=cap(P_0) + cap(P_1) + \ldots + cap(P_n) = \log_2 |P_0| + cap(\mathcal{M}).$$

Thus, in consequence of Theorem 2.4, we obtain

(2.28) $$\log_2 |P_0| + cap(\mathcal{M}) = cap(\mathcal{M}') \leq \log_2 |\mathcal{S}| .$$

(2.26),(2.28) yield

$$\log_2 |P| + \log_2 |\mathcal{S}| - 1 < \log_2 |P| + cap(\mathcal{M}) \leq \log_2 |\mathcal{S}|,$$

whence

$$\log_2 |P| < 1 \quad \text{i.e.} \quad |P| < 2$$

which is impossible by $\mathcal{M}_0 \in UMEM(\mathcal{S})$. ■

The following example shows that an independent memory can be maximal without the fulfilment of condition (2.26).

EXAMPLE 2.2. Let $\mathcal{S}$ be a low segment of non-negative integers such that $|\mathcal{S}| > 7$, otherwise, the cardinality of $\mathcal{S}$ is irrelevant, it can be finite or infinite: $\mathcal{S} = \{0,1,2,3,4,5,6,7,\ldots\}$.

We will construct a 3-bit memory with two memory cells. The generator fuctions $f:\mathbb{S} \to \{0,1,2,3\}$ and $g:\mathbb{S} \to \{0,1\}$ are defined by means of the following table:

n	0	1	2	3	4	5	6	7	8	. . .	(7≤)k	. . .
f(n)	0	1	2	3	0	1	2	3	3	. . .	3	. . .
g(n)	0	0	0	0	1	1	1	1	1	. . .	1	. . .

FIGURE 1

Let P_1 and P_2 be the partitions generated by f and g, respectively. Then

$$P_1 = \{\{0,4\},\{1,5\},\{2,6\},\{3,7,8,9,...\}\},$$
$$P_2 = \{\{0,1,2,3\},\{4,5,6,7,8,9,...\}\}$$

First we show that the memory

$$\mathcal{M} = (\{0,1,2,3,4,5,6,7,...\},\{P_1,P_2\},s)$$

is independent. This follows from Corollary 2.2, since

$f^{-1}(0) \cap g^{-1}(0) = \{0\}$, $f^{-1}(0) \cap g^{-1}(1) = \{4\}$, $f^{-1}(1) \cap g^{-1}(0) = \{1\}$, $f^{-1}(1) \cap g^{-1}(1) = \{5\}$, $f^{-1}(2) \cap g^{-1}(0) = \{2\}$, $f^{-1}(2) \cap g^{-1}(1) = \{6\}$, $f^{-1}(3) \cap g^{-1}(0) = \{3\}$, $f^{-1}(3) \cap g^{-1}(1) = \{7,8,9,...\}$,

and thus, the condition (2.15) is fulfilled.
In order to prove the maximality of $\mathcal{M}$, it suffices to show the followings: if we suppose the existence of a non-constant function $h:\mathbb{S} \to \mathbb{H}$ for which the memory $\mathcal{M}' = (\mathbb{S},\langle\{f,g,h\}\rangle,s)$ is independent and the functions f,g,h are strong different then we obtain a contradiction.

Since h is non-constant, there exist $y_1, y_2 \in \mathbb{H}$ such that

(2.29) $$y_1 \neq y_2 .$$

Since $\mathcal{M}'$ is independent, on the basis of Corollary 2.2, we have $\{0\} \cap h^{-1}(y_1) = f^{-1}(0) \cap g^{-1}(0) \cap h^{-1}(y_1) \neq \emptyset$ and similarly, $\{0\} \cap h^{-1}(y_2) \neq \emptyset$. So, $0 \in h^{-1}(y_1) \cap h^{-1}(y_2)$ whence $y_1 = h(0) = y_2$ which contradicts (2.29). Thus, $\mathcal{M}$ is maximal. Since $\mathcal{M}$ is independent, on the basis of Theorem 2.4, the capacity of $\mathcal{M}$ is equal to the sum of capacities of the memory cells of $\mathcal{M}$:

$$cap(\mathcal{M}) = cap(P_1) + cap(P_2) = \log_2 |P_1| + \log_2 |P_2| = \log_2 4 + \log_2 2 = 3.$$

2.5. The vector cell of a memory

We continue the investigations of the maximality of independent memories. Using the notion of class intersections of a memory, we can define the vector cell of a memory $\mathcal{M}$ even if $\mathcal{M}$ is infinite. Several properties of a memory $\mathcal{M}$ may be characterized by means of the properties of the vector cell of $\mathcal{M}$. We give here criteria for the independence and the maximality of a memory.

DEFINITION 2.11. Let $\mathcal{M}$ be a pure memory defined on a set S. Let exactly one class be chosen from every partition of S belonging to the core of $\mathcal{M}$. The intersection of all selected classes will be called a class intersection of $\mathcal{M}$.

In the formula (2.15) of Corollary 2.2, speaking, the class intersections play a role. In the next theorem, we generalize this result omitting the requirement of finiteness of $\mathcal{M}$.

THEOREM 2.6. A pure memory $\mathcal{M}$ is independent if and only if neither of the class intersections of $\mathcal{M}$ is empty.

Proof. Necessity. Suppose that there exixts an empty class intersection of the memory $\mathcal{M} = (S, C, s)$. Then, necessarily, $\mathcal{M}$ is multicellular, since every class intersection of a unicellular memory is a class /of a single partition of S/ which can't be empty. For all $P \in C$, we denote by $cl(P)$ the class of P which takes part in the forming of the empty class intersection which is in question. Thus,

$$\bigcap_{P \in C} cl(P) = \emptyset \tag{2.30}$$

We shall show that neither memory cell P of $\mathcal{M}$ is independent in $\mathcal{M}$. Indeed, we have $\forall P \in C$

$$\forall s'' \in cl(P)\ \exists Q \in C : s'' \notin cl(Q) \tag{2.31}$$

since in the opposite case $\exists P \in C\ \exists s'' \in cl(P)\ \forall Q \in C : s'' \in cl(Q)$ and so, the intersection of the classes $cl(Q)$ would not empty. For some $P \in C$, let a Q be fixed which is declared in (2.31) and let f and g be the corresponding generator functions. Then, for all $s' \in cl(P)$ and $s \in cl(Q)$, it is impossible to exchange $cont(P)|s = f(s)$ to $cont(P)|s'' = f(s'') = f(s')$ without disturbance the contents of the other cells in $\mathcal{M}$. Namely, for all $s'' \in cl(P)$, $s'' \notin (g)s$, i.e. $g(s'') \neq g(s)$ and so $cont(Q)|s'' = g(s'') \neq g(s) = cont(Q)|s$. Thus, we have proved that if $\mathcal{M}$ has an empty class intersection then $\mathcal{M}$ can't be independent. So, we have proved the necessity of the condition.

Sufficiency. Let $\mathcal{M} = (S, \langle F \rangle, s)$ be a memory with non-empty class intersections. So, condition (2.11) is fulfilled and thus, by Corollary 2.1, $\mathcal{M}$ is independent. ■

LEMMA 2.3. Let $\mathcal{M}$ be a pure memory on S. Then (i) the set of all non-empty class intersections of $\mathcal{M}$ form a partition P_0 of S, (ii) if $\mathcal{M}$ is finite then P_0 is the vector cell of $\mathcal{M}$.

Proof. (i) According to the definition of P_0, the elements of P_0 are non-empty subsets of S. We shall show that the union of the sets of P_0 is S. Every $s \in S$ is contained in exactly one class of every partition of S. Taking the intersection of all such classes of the memory cells of $\mathcal{M}$, we obtain the class intersection which contains the element s. Thus, the union of all class intersections contains every element of S. In this way, we have proved that the union of the sets of P_0 is S. Finally, it remains to prove that the intersection of two different class intersections is empty. Indeed, if S' and S'' are different elements of P_0 then there exixts a memory cell P of $\mathcal{M}$ which has two different classes S_1 and S_2 such that S_1 takes part in the forming of the class intersection S' and S_2 takes part in the forming of S'': $S' = \ldots \cap S_1 \cap \ldots$ and $S'' = \ldots \cap S_2 \cap \ldots$. Thus, since $S_1 \cap S_2 = \emptyset$, we get $S' \cap S'' = \emptyset$.
(ii) Let $\mathcal{M} = (S, \{f_1, \ldots, f_n\}, s)$ be a finite memory, where the generator functions $f_i : S \to H_i$ $/i \in \{1, \ldots, n\}/$ are strong different. Every class intersection may be written in the form

$$f_1^{-1}(h_1) \cap \ldots \cap f_n^{-1}(h_n) \quad \text{where} \quad h = (h_1, \ldots, h_n) \in H_1 \times \ldots \times H_n.$$

As we have seen by the proof of Corollary 2.2, the condition $s \in f_1^{-1}(h_1) \cap \ldots \cap f_n^{-1}(h_n)$ is equivalent to the property $h = F_0(s)$ which is equivalent to $s \in F_0^{-1}(h)$. This shows that $f_1^{-1}(h_1) \cap \ldots \cap f_n^{-1}(h_n) = F_0^{-1}(h)$ and thus F_0 is a generator function of P_0 i.e. P_0 is the vector cell of $\mathcal{M}$. ■

DEFINITION 2.12. Let $\mathcal{M}$ be a pure memory on a set S. By the vector cell of $\mathcal{M}$, we mean the partition P_0 consisting of the non-empty class intersections of $\mathcal{M}$.

THEOREM 2.7. An independent memory $\mathcal{M}$ is maximal iff the vector cell of $\mathcal{M}$ has a class consisting of a single element.

Proof. Sufficiency. Let $\mathcal{M} = (S, C, s)$ be an independent memory and let $\{s_0\}$ be a one element class of the vector cell of $\mathcal{M}$. In order to prove the maximality of $\mathcal{M}$, on the basis of Lemma 2.1, it suffices to show that neither extension of $\mathcal{M}$ with a unicellular memory is independent. Let $\mathcal{M}_1 = (S, \{P\}, s)$ be a pure unicellular memory such that P is not a memory cell of $\mathcal{M}$ and thus, $\mathcal{M}' = \mathcal{M}_1 \cup \mathcal{M}$ is an extension of $\mathcal{M}$: $\mathcal{M}' \supset \mathcal{M}$. Since $\mathcal{M}_1$ is a pure memory, the partition P is proper i.e. P contains at least two different classes K_1 and K_2. So,

$$K_1 \cap K_2 = \emptyset. \tag{2.32}$$

Both $\{s_0\} \cap K_1$ and $\{s_0\} \cap K_2$ are class intersections of $\mathcal{M}'$ and at least one of these class intersections is empty. Namely, the opposite case yields $s_0 \in K_1 \cap K_2$ which is impossible by (2.32). Thus, according to Theorem 2.6, $\mathcal{M}'$ is not independent and by Lemma 2.1, $\mathcal{M}$ is maximal.
Necessity. Let $\mathcal{M} = (S, C, s)$ be an independent memory such that every class intersection of $\mathcal{M}$ has more then one element. Let $P_0 = \{S_n \mid n \in \Gamma\}$ be the vector cell of $\mathcal{M}$. For all $n \in \Gamma$ let an element

$$s_n \in S_n \tag{2.33}$$

be chosen. Then, in consequence of our supposition, $\forall n \in \Gamma$

(2.34) $$S'_n = S_n \setminus \{s_n\} \neq 0 \quad .$$

Let the sets T_0, T_1 be defined as follows:

(2.35) $$T_0 = \{s_m \mid m \in \Gamma\},$$

(2.36) $$T_1 = \bigcup_n S'_n \quad .$$

We shall show that $\{T_0, T_1\}$ is a partition of $\mathbb{S}$. $T_0 \neq 0$ in consequence of (2.33) and (2.35). $T_1 \neq 0$ in consequence of (2.34) and (2.36). Next we show that

(2.37) $$T_0 \cap T_1 = 0 \quad .$$

Suppose that (2.37) is false. Then $\exists\, s \in \mathbb{S}$:

(2.38) $$s \in T_0$$

and

(2.39) $$s \in T_1 \quad .$$

It follows from (2.38) that $\exists\, m \in \Gamma$: $s = s_m$ and from (2.39) that $\exists\, n \in \Gamma$: $s \in S'_n$ and so we have

(2.40) $$s_m \in S'_n \quad .$$

It follows from (2.33) and (2.40) that $s_m \in S_m \cap S_n$ from which we get $m = n$ /since the vector cell of $\mathcal{M}$ is a partition of $\mathbb{S}$/. But, by (2.40), this leads to the contradiction: $s_n \in S'_n = S_n \setminus \{s_n\}$. This shows that (2.37) is true. It is still left to show that

(2.41) $$T_0 \cup T_1 = \mathbb{S} \quad .$$

Indeed,

$$T_0 \cup T_1 = \{s_n \mid n \in \Gamma\} \cup \bigcup_n (S_n \setminus \{s_n\}) = \bigcup_n S_n = \mathbb{S} \quad .$$

Thus, we have proved that $\{T_0, T_1\}$ is a proper partition of $\mathbb{S}$. Now, we shall prove that

(2.42) $$\{T_0, T_1\} \notin \mathbb{C}$$

Suppose that $\{T_0, T_1\} \in \mathbb{C}$. Then $\exists\, n \in \Gamma$: $S_n = T_1 \cap Q$ where Q is the intersection of some classes selected from other cells of $\mathbb{C}$. Whence, we get $s_n \in T_1$ which is impossible, since $s_n \in T_0$ and the classes T_0 and T_1 are disjoint sets. Thus, (2.42) is true and so $\mathcal{M}' = \mathcal{M}_1 \cup \mathcal{M}$ is an extension of $\mathcal{M}$ in case $\mathcal{M}_1 = (\mathbb{S}, \{\{T_0, T_1\}\}, s)$. Next, we prove that $\mathcal{M}' = \mathcal{M}_1 \cup \mathcal{M}$ is independent. In consequence of (2.42), the class intersections of $\mathcal{M}_1 \cup \mathcal{M}$ may be written in the form

(2.43) $$T_0 \cap S_n \qquad /n \in \Gamma/ \quad \text{and}$$

(2.44) $$T_1 \cap S_n \qquad /n \in \Gamma/ \quad .$$

The class intersections of the form (2.43) are not empty since by (2.33) and (2.35), $s_n \in T_0 \cap S_n$. The class intersections (2.44) are not empty, since in consequence of (2.36) and (2.34) we have

$$T_1 \cap S_n = \left(\bigcup_m S'_m\right) \cap S_n = S'_n \neq 0 \quad .$$

Thus, by Theorem 2.6, $\mathcal{M}' = \mathcal{M}_1 \cup \mathcal{M}$ is independent. So, $\mathcal{M}'$ is an independent extension of $\mathcal{M}$, therefore, $\mathcal{M}$ cannot be maximal. Consequently, the condition that the vector the vector cell of $\mathcal{M}$ has a one element class, is a necessary condition for the maximality of $\mathcal{M}$. ■

2.6. Economically defined memories

It may happen related to a memory that there exist different states s_1, s_2 such that the distributions of the contents among the memory cells at the states s_1 and s_2 are identical. Then, at least one of s_1 and s_2 may be regarded as a superfluous state of the memory. But, according to an other consideration, we will say in this case that one handled not quite economically with the states by the construction of the memory.

One can think that an economically defined memory has some excellent properties, in particular concerning the capacity. This will be the content of this chapter.

DEFINITION 2.13. Let $\mathcal{M} = (S, C, s)$ be a pure memory.
(i) The states $s_1, s_2 \in S$ are called undistinguishable by the memory $\mathcal{M}$, denoted by $s_1 (\mathcal{M}) s_2$, if the distributions of the contents among the memory cells of $\mathcal{M}$ are identical at the states s_1 and s_2, formally,

(2,45) $$s_1 (\mathcal{M}) s_2 \iff \forall P \in C : cont(P)|s_1 = cont(P)|s_2 .$$

(ii) The element $s_0 \in S$ is said to be an essential state of $\mathcal{M}$ if for all $s \in S$ the states s and s_0 are undistinguishable by $\mathcal{M}$ only in the trivial case $s = s_0$.

The following definition takes its origiin from the colour recognition /[4]/.

DEFINITION 2.14. The element $s_0 \in S$ is said to be recognizable by the function $f: S \to H$ if $\forall s \in S : s(f)s_0 \Longrightarrow s = s_0$.

THEOREM 2.8. Let $\mathcal{M}$ be a pure memory on the finite set S. Then

(i) $$(\mathcal{M}) \leftrightarrow (F_0)$$

where F_0 is a generator function of the vector cell of $\mathcal{M}$.
(ii) The element $s_0 \in S$ is an essential state of $\mathcal{M}$ if and only if s_0 is recognizable by F_0.

Proof. Let $C = \{P_1, \ldots, P_n\}$ be the core of $\mathcal{M}$ and $f_1, \ldots, f_n$ be the generator functions of $P_1, \ldots, P_n$, respectively. Then the distributions of the contents among the memory cells of C at the states s_1, s_2 are given by the vectors $F_0(s_1)$, $F_0(s_2)$, respectively, where $F_0(s) = [f_1(s), \ldots, f_n(s)]$. Thus, by (2.45),

$$s_1(\mathcal{M})s_2 \iff \forall P \in C : cont(P)|s_1 = cont(P)|s_2 \iff \forall i \in \Gamma_n : f_i(s_1) = f_i(s_2) \iff$$
$$\iff F_0(s_1) = F_0(s_2) \iff s_1 (F_0) s_2$$

which proves the statement (i). The statement (ii) is a simple consequence of (i) and the definitions 2.13 and 2.14. ■

DEFINITION 2.15. A memory $\mathcal{M}$ is said to be economically defined on S if (i) $\mathcal{M}$ is independent and (ii) every element of S is an essential state of $\mathcal{M}$.

THEOREM 2.9. Let $\mathcal{M}$ be an independent memory on a finite set S. Then, the following statements (I) and (II) are equivalent.
(I) $\mathcal{M}$ is economically defined on S.
(II) The capacity of $\mathcal{M}$ is the largest possible on S:

$$\text{cap}(\mathcal{M}) = \log_2 |S| \ . \tag{2.46}$$

Proof. (II) $\Longrightarrow$ (I) . It follows from (2.46) by Proposition 2.2 that any generator function of the vector cell of $\mathcal{M}$ is injective. Denoting by F_0 a generator function of the vector cell of $\mathcal{M}$, by Theorem 2.8 (i) and Remark 1.1, we get

$$\forall\, s\in S\ \forall\, s'\in S\ :\ s(\mathcal{M})s' \iff s(F_0)s' \iff s = s'$$

which shows that every $s\in S$ is an essential state of $\mathcal{M}$ and thus, the memory $\mathcal{M}$ is economically defined on S.
(I) $\Longrightarrow$ (II). Clearly, it suffices to prove that $\neg$(II) $\Longrightarrow$ $\neg$(I). In order to prove this, suppose that (2.46) is not true. Then, by Theorem 2.4,

$$\text{cap}(\mathcal{M}) < \log_2 |S| \ .$$

Thus, by Definition 2.7,

$$\log_2 |F_0(S)| = \text{cap}(\mathcal{M}) < \log_2 |S| ,$$

whence, $|F_0(S)| < |S|$ showing that F_0 is not an injection. Thus, there exist elements $s_1 \neq s_2$ of S for which $F_0(s_1)=F_0(s_2)$. Therefore, on the basis of Theorem 2.8, we obtain

$$F_0(s_1) = F_0(s_2) \iff s_1(F_0)\, s_2 \iff s_1(\mathcal{M})\, s_2 \ .$$

This proves that $\mathcal{M}$ can't be economically defined on S because not all elements of S are essential states of $\mathcal{M}$, since at least s_1 and s_2 are surely undistinguishable by $\mathcal{M}$. ■

REMARK 2.11. There exists an economically defined memory on any finite set S provided $|S| > 1$, as this shows the simple example of the unicellular memory $\mathcal{M}_1=(S,\{P\},s)$ where $P = \{\{s\} \mid s\in S\}$. Namely, by Theorem 2.1, $\mathcal{M}_1$ is independent /since P is a proper partition of S in consequence of $|S|>1$ /, moreover,

$$\text{cap}(\mathcal{M}_1) = \log_2 |P| = \log_2 |S|$$

and thus, by Theorem 2.9, $\mathcal{M}_1$ is economically defined on S .

Next, we give a nontrivial example for a multicellular economically defined memory which takes its origin from the practice.

The memory, described below, is used in the practice as the operative memory of a middle-sized personal computer.

EXAMPLE 2.3. Let $\mathbb{B} = \{0,1,2,\ldots,255\}$ be a low segment of integers and let $\mathbb{S}$ be the Cartesian product

$$\mathbb{S} = \mathbb{B}^{65536} = \overset{1}{\mathbb{B}} \times \overset{2}{\mathbb{B}} \times \ldots \times \overset{65536}{\mathbb{B}} .$$

Thus we have

$$|\mathbb{S}| = 256^{65536} ,$$

which is practically infinite /the number of atoms in the observable universe is not more then 10^{100} /. However, the largest possible capacity of a memory which can be defined on $\mathbb{S}$ is not too much since

(2.47) $$\log_2 |\mathbb{S}| = 65536 . \log_2 256 = 524288 \text{ bits} = 65 \text{ kbytes}.$$

This is the capacity of the memory /ROM U RAM/ of a ZX-Spectrum or a Commodore 64, etc. personal computer.

The elements of $\mathbb{S}$ are the long vectors: $s = (s_0, s_1, \ldots, s_m)$ where $m=65535$ and $\forall\, n \in \{0,\ldots,m\}$: $s_n \in \mathbb{B}$. The generator functions of the memory, which is to describe, are the base functions $f_n : \mathbb{S} \to \mathbb{B}$ defined in the following way: Let $\Gamma_m = \{0,1,\ldots,m\}$ and

(2.48) $$\forall n \in \Gamma_m \; \forall s \in \mathbb{S} : f_n(s) = s_n , \quad \text{for } s=(s_0,\ldots,s_n,\ldots,s_m).$$

Let $P_n = f_n^{-1}(\mathbb{B})$ be the memory cell generated by f_n /$n \in \Gamma_m$/.
We consider the memory

(2.49) $$\mathcal{M} = (\mathbb{S}, \{P_0, P_1, \ldots, P_m\}, s)$$

It is easy to check that the class intersection of $\mathcal{M}$ determined by the vector $h = (h_0, h_1, \ldots, h_m) \in \mathbb{S}$, is the one element set

(2.50) $$f_0^{-1}(h_0) \cap f_1^{-1}(h_1) \cap \ldots \cap f_m^{-1}(h_m) = \{(h_0, h_1, \ldots, h_m)\}$$

Thus, the vector cell of $\mathcal{M}$ is

(2.51) $$P_0 = \{\{s\} \mid s \in \mathbb{S}\} .$$

Consequently, by Theorem 2.6, $\mathcal{M}$ is independent and by Theorem 2.7, $\mathcal{M}$ is maximal.
It follows from (2.51) that $|P_0| = |\mathbb{S}|$ and so, according to the calculation (2.47), the capacity of $\mathcal{M}$ is 65 kbytes. This is the largest possible capacity of a memory on the set $\mathbb{S}$ and thus, by Theorem 2.9, $\mathcal{M}$ is economically defined on $\mathbb{S}$.

3. INFORMATION

The capacity of memory and the measure of information

The capacity of memory is one and the same thing as the measure of information. Namely, according to the determination used in the systems programing, " memory is the device where information is stored" /[2]/ Thus, if we have introduced the measure of information then we can define the capacity of memory by means of the quantity of information, which is stored in the memory. Inversely, if the capacity of memory is regarded as the primary concept then we can measure the information by means of the capacity of that part of the memory which is needed to store the information.

In the theory of information, there are known several approaches to the measures of information. According to a well known paper of A.N. KOLMOGOROV [9], we may speak about the following approaches:

1. The combinatorical approach, where the measure of information, (the entropy of the variable x on a finite set X), is

(H) $$H(x) = \log_2 |X|$$

2. The probabilistic approach, where the Shannon entropy

(S) $$H_w(x) = - \sum_x p(x) \log_2 p(x)$$

is the measure of information and here, x is a random variable.

3. The algorithmic approach / see [9] / .

In the last two decads, a new branch of the information theory developed by the activity of many authors / see the book [1] of J. ACZÉL and Z. DARÓCZY as well as the recent work [11]/. The trend of this branch is to characterize the Shannon entropy axiomatically in terms of some natural properties which are essential from the point of view of information theory.

It is easy to see by Definition 2.7, that our definition of the capacity of memory may be regarded as the combinatorical entropy of a class of the vector cell of the memory.

The origin of the combinatorical entropy goes back in 1928 to R. V. HARTLEY / [7] /. Therefore, we will name the combinatorical entropy from Hartley. According to a widespread view, the Hartley entropy may be considered as a special case of the Shannon entropy. The cause for such conclusion gives the fact that for the uniform probability distribution, the formula (S) of Shannon's entropy reduces to the formula of Hartley's entropy. But, in the present case, it seems to be unnatural to force the view of stochastic treatment, since it is unjustified to speak about some kind of probability distribution in connection with the capacity of a memory.

We have introduced the notion of the capacity of memory in three steps. First, in Definition 2.2, we defined the capacity of a memory cell P / which is a partition of a set S of states/

as the logarithm of the cardinality of the partition P:

$$(*) \qquad cap(P) = \log_2 |P| \ .$$

We can speak about the content of P only in relation to the instant state $s \in S$ of the memory. Let $\mathcal{M} = (S, C, s)$ be a memory /where C, called the core of $\mathcal{M}$, is a set of partitions of S / and let $P = \{K_0, K_1, \ldots, K_{N-1}\} \in C$. Since P is a partition of S, any instant state $s \in S$ is contained in some class $K_n \in P$, $n \in \Gamma_N = \{0,1,\ldots,N-1\}$. We say in this case, that the number n is the content of P at state s. Formally:

$$(**) \qquad cont(P)|s=n \iff s \in K_n \ .$$

Now, if one intends to use the probabilistic approach to the measure of information, then one can define the capacity of P as the entropy of the event $s \in K_n$. But, in order to obtain (*), it is needed to know that any event $s \in K_n$ has a probability p_n such that $p_0 = p_1 = \ldots = p_{N-1} = 1/N$.

On the other hand, we have a good motivation for (*) without reference to probability theory. We choose a fixed number system, say, the binary system, which is used frequently in the computer technics. The capacity of a memory cell will be determined thereby, that how long numbers n can be stored in the memory cell. The length of n is the number of digits in the binary form of n. Thus, for example in case of $|P|=2^m$, a number n can be stored in P provided, $0 \leq n \leq 2^m-1$. The length of n may be obtained from the binary form

$$n = a_0 + a_1 2 + \ldots + a_{m-1} 2^{m-1} \ .$$

The maximal length of such numbers is $m = \log_2 |P|$ which will be measure the capacity of P.

In the second step, we defined the capacity of the core $C = \{P_1, \ldots, P_n\}$. We have seen that this case is not so simple that we could take the sum of capacties of the memory cells of C . This goes only in case of independence.

In the third step, we defined the capacity of the memory as the capacity of its core.

The aim of this chapter is to characterize the capacity of memory axiomatically in terms of some properties which are natural from the point of view of computer technics. Naturally, in this way, we get a characterization of the Hartley entropy too, at the least formally. It may happen that the properties, which will be used for the characterization of the capacity of memory, are not so natural from the view of point of the information theory.

But, a large difference cannot exist between the two points of view. Roughly speaking, the information theory deals with the transmission of information whilst we deal here in the processor theory with the storage of information. However, as we shall see later / not in this chapter/ , that the transmission of information may be thought of as a writing and reading in a memory /which will be the channel/. Inversely, any writing and reading in a memory is connected with a transmission of information.

3.1. The semi group of partitions

We shall denote by S a set for which $2 \le |S| \le \infty$. The set of all finite partitions of S will be denoted by $P_1(S)$. We denote by $P(S)$ the subset of $P_1(S)$ which consists of all proper partitions of S. Thus, $P \in P(S) \iff P \in P_1(S) \wedge |P| \ge 2$, moreover,

$$P_1(S) = P(S) \cup \{E\}$$

where $E=\{S\}$ is the improper partition of S. Here, the interesting set will be $P(S)$ since the improper partition is unsuited for the storage of information.

In this section, we shall define a multiplication of partitions and we shall deal with some properties of the introduced operation.

DEFINITION 3.1. By the product $P.Q$ of the partitions P,Q of S, we mean the set of all nonempty class intersections of P and Q :

$$(3.1) \quad P.Q = \{S \subset S \mid S = K \cap L \text{ if } K \in P, L \in Q \text{ and } K \cap L \neq \emptyset\}.$$

We remind that P and Q are independent iff all class intersections of $\{P,Q\}$ are nonemty sets / Theorem 2,6 / .

PROPOSITION 3.1. Let $P,Q \in P(S)$, Then, we have (i)

$$(3.2) \quad \max\{|P|,|Q|\} \le |P.Q| \le |P|.|Q|$$

and (ii) P and Q are independent if and only if

$$(3.3) \quad |P.Q| = |P|.|Q| .$$

Proof. Let the arbitrary elements $P,Q \in P(S)$ be written in the form $P = \{K_0,\dots,K_N\}$, $Q = \{L_0,\dots,L_M\}$ where $N,M \ge 1$ because P and Q are proper partitions of S. Since the classes of a partition are nonempty sets and the union of classes of a partition of S is S, we have $\forall n \in \Gamma_N : \emptyset \neq K_n = K_n \cap S = K_n \cap (L_0 \cup \dots \cup L_M) = (K_n \cap L_0) \cup \dots \cup (K_n \cap L_M)$ whence $\forall n \in \Gamma_N \; \exists m \in \Gamma_M : K_n \cap L_{m(n)} \neq \emptyset$. Thus, for all $n \in \Gamma_N$, we can choose an $m(n) \in \Gamma_M$ such that $K_n \cap L_{m(n)} \neq \emptyset$. Then,

$|P| = N + 1 = |\{K_n \cap L_{m(n)} \mid n \in \Gamma_N\}| \le |P.Q|$ and similarly, $|Q| \le |P.Q|$, which prove the first inequality of (3.2). We know that neither of the class intersections is empty exactly in case of independence of P and Q. Thus, the equality

$$|P.Q| = |\{K_n \cap L_m \mid n \in \Gamma_N, m \in \Gamma_M\}| = (N+1)(M+1) = |P|.|Q|$$

holds if and only if P,Q are independent and part (ii) is proved. If P,Q are dependent then some class intersections are void. So,

$$|P.Q| < |\{K_n \cap L_m \mid n \in \Gamma_N, m \in \Gamma_M\}| = (N+1)(M+1) = |P|.|Q|$$

which together with (3.3) yield the second inequality of (3.2). ■

An element a of a semi group is said to be idempotent if $a.a = a$. If all elements of a semi group are idempotent then we will say that the semi group is idempotent.

THEOREM 3.1. The multiplication (3.1) is a binary operation on the set $\mathbb{P}(S)$. With respect to this operation, $\mathbb{P}(S)$ forms an idempotent commutative semi group.

Proof. We have proved in Lemma 2.2, that the set of all nonempty class intersections of partitions belonging to an arbitrary collection of partitions of S forms a partition of S. Therefore, by Definition 3.1, $P.Q$ is a partition of S for all $P, Q \in \mathbb{P}(S)$. Moreover, in consequence of Proposition 3.1,

$$|P.Q| \geq \max\{|P|,|Q|\} \geq |P| \geq 2$$

since $|P| \geq 2$ for all $P \in \mathbb{P}(S)$. Thus, $P.Q$ is proper and so $P.Q \in \mathbb{P}(S)$. Since the intersection of sets is a commutative and associative operation, we see from (3.1) that $P.Q = Q.P$ and $(P.Q).R = P.(Q.R)$ for all $P,Q,R \in \mathbb{P}(S)$. Thus, we have proved that $\mathbb{P}(S)$ is a commutative semi group. Finally, we prove the idempotence of an arbitrary $P \in \mathbb{P}(S)$. This is a simple consequence of the fact that the classes of a partition are nonempty sets and the intersection of two different classes is empty. Thus, we have

$$P.P = \{K_0,\dots,K_N\}.\{K_0,\dots,K_N\} = \{K_0 \cap K_0,\dots,K_N \cap K_N\} = \{K_0,\dots,K_N\} = P. \blacksquare$$

REMARK 3.1. It is easy to see that the unit element of the multiplication (3.1) is the improper partition $E=\{S\}$. So, $\mathbb{P}_1(S)$ is an idempotent commutative semi group with unit element.

REMARK 3.2. Let $C=\{P_1,\dots,P_n\}$ be the core of the memory $\mathcal{M}$ on S. As we can see easily, the vector cell of $\mathcal{M}$ is not other than the product $P_1 \dots P_n$. Using this fact and the above theorem, we can lightly clear some former statements. For example, we stated in Remark 2.7 without any proof that the definition of the capacity of the memory / Definition 2.7 / is correct in the sense that $\mathrm{cap}(\mathcal{M}) = \log_2 |\mathbb{P}_0(S)|$ does not depend of the permutations of $P_1,\dots,P_n$. Here, $\mathbb{P}_0$ is the generator function of the vector cell of $\mathcal{M}$ and thus, $|\mathbb{P}_0(S)|$ is the cardinality of the vector cell. Therefore, $\mathrm{cap}(\mathcal{M}) = \log_2 |P_1 \dots P_n|$. So, it is clear, that $P_1 \dots P_n$ is insensible to the permutations of $P_1,\dots,P_n$ in consequence of the commutativity and associativity of the multiplication of partitions. Thus, Definition 2.7 is correct.

One can prove by induction the following generalization of Proposition 3.1.

THEOREM 3.2. Let $P_1,\dots,P_n \in \mathbb{P}(S)$. Then (i)

$$\max\{|P_1|,\dots,|P_n|\} \leq |P_1 \dots P_n| \leq |P_1| \dots |P_n| \quad , \tag{3.4}$$

(ii) $\{P_1,\dots,P_n\}$ is independent if and only if

$$|P_1 \dots P_n| = |P_1| \dots |P_n| \; . \tag{3.5}$$

3.2. The size of the core

In this chapter, by a core we shall mean a finite non-void subset of $P(S)$. The set of all such subsets of $P(S)$ will be denoted by $\mathcal{C}(S)$.

DEFINITION 3.2. Let $C = \{P_1,\dots,P_N\} \in \mathcal{C}(S)$ be a core.
(i) By the vector cell of the core C , we mean the product

$$C^{\wedge} = P_1 \dots P_N \ . \tag{3.6}$$

(ii) By the size of the core C we mean the cardinality of the vector cell of C . The size of C will be denoted by $\|C\|$. So,

$$\|C\| = |C^{\wedge}| = |P_1 \dots P_N| \ . \tag{3.7}$$

REMARK 3.3. In general, there is no connection between the size $\|C\|$ and the cardinality $|C|$ of the core.

THEOREM 3.3. Let $C, C_1, C_2 \in \mathcal{C}(S)$ be given arbitrarily. Then
(i)

$$\|C\| \geq 2. \tag{3.8}$$

(ii) $C = \{P_1,\dots,P_N\}$ is independent if and only if

$$\|C\| = |P_1| \dots |P_N| \ . \tag{3.9}$$

(iii)

$$C_1 \subset C_2 \Rightarrow \|C_1\| \leq \|C_2\| \ , \tag{3.10}$$

If C_2 is independent then

$$C_1 \subset C_2 \Rightarrow \|C_1\| < \|C_2\| \ . \tag{3.11}$$

(iv)

$$\|C_1 \cup C_2\| \leq \|C_1\| . \|C_2\| \ . \tag{3.12}$$

Proof. (i) (3.8) follows immediately from (3.7) and (3.2) .
(ii) This is an immediate consequence of Theorem 3.2.(ii)
(iii) Let $C_1 = \{P_1,\dots,P_N\}$ and $C_2 = \{P_1,\dots,P_N,Q_1,\dots Q_M\}$. Then, by Theorem 3.1, Proposition 3.1,

$$\|C_2\| = |P_1 \dots P_N . Q_1 \dots Q_M| = |(P_1 \dots P_N)(Q_1 \dots Q_M)| \geq |P_1 \dots P_N| = \|C_1\|$$

and (3.10) is proved. If C_2 is independent then any subcore of C_2 is also independent. We show that $\|C_1\| = \|C_2\|$ is impossible. Namely, by (ii) , we get

$$|P_1| \dots |P_N| = \|C_1\| = \|C_2\| = |P_1| \dots |P_N| . |Q_1| \dots |Q_M|$$

whence $1 = |Q_1| \dots |Q_M|$ which is impossible in consequence of $Q_n \in P(S)$ if $n \in \{1,\dots,M\}$. Thus, $\|C_1\| < \|C_2\|$ by (3.10) .

(iv) We distinguish two cases a: $C_0 = C_1 \cap C_2 = \emptyset$, b: $C_0 \neq \emptyset$.
(a) In this case, if $C_1 = \{P_1,\ldots,P_n\}$, $C_2 = \{Q_1,\ldots,Q_m\}$ then $C_1 \cup C_2 = \{P_1,\ldots,P_n,Q_1,\ldots,Q_m\}$. So by Theorem 3.1, Proposition 3.1.(i), we get
$\|C_1 \cup C_2\| = |(P_1,\ldots,P_n)(Q_1,\ldots,Q_m)| \leq |P_1\ldots P_n|\,|Q_1\ldots Q_m| = \|C_1\|\,\|C_2\|$.

(b) In this case, $C_1 \cap C_2 = \{R_1,\ldots,R_N\} \neq \emptyset$ and $C_1 = \{P_1,\ldots,P_n,R_1,\ldots,R_N\}$, $C_2 = \{Q_1,\ldots,Q_m,R_1,\ldots,R_N\}$, where, if $C_1 = C_0$ then $n=1$, $P_1=E$ and if $C_2 = C_0$ then $m=1$, $Q_1=E$. Thus, in any case, $|Q_1\ldots Q_m| \leq |Q_1\ldots Q_m R_1\ldots R_N| = \|C_2\|$ and so, $\|C_1 \cup C_2\| = |(P_1\ldots P_n R_1\ldots R_N)(Q_1\ldots Q_m)| \leq \|C_1\|\,|Q_1\ldots Q_m| \leq \|C_1\|\,\|C_2\|$. ■

A core C_1 may be dependent upon the core C_2 even if $C_1 \cup C_2$ is independent. Namely, if $C_1 \cap C_2 \neq \emptyset$ then every alteration of the content in a memory cell $R \in C_1 \cap C_2$ results the same alteration in both C_1 and C_2.

DEFINITION 3.3. The core C_1 is said to be independent of the core C_2 if $C_1 \cup C_2$ is independent and $C_1 \cap C_2 = \emptyset$.

REMARK 3.4. It is clear that if $C_1 \cup C_2$ is independent then both C_1 and C_2 are independent individually. But, the inverse of the above statement is not true, as that simple examples show.
Thus, the following implications hold:
"each of C_1, C_2 is independent of the other" $\Rightarrow$ "$C_1 \cup C_2$ is independent" $\Rightarrow$ "both C_1 and C_2 are independent individually".

THEOREM 3.4. If $C_1 \in \mathcal{C}(S)$ is independent of $C_2 \in \mathcal{C}(S)$ then

(3.13) $$\| C_1 \cup C_2 \| = \|C_1\| . \|C_2\| .$$

Conversely, if both C_1 and C_2 are independent individually then the fulfilment of (3.13) implies that each of C_1, C_2 is independent of the other.

Proof. Suppose that each of $C_1=\{P_1,\ldots,P_N\}$ and $C_2=\{Q_1,\ldots,Q_M\}$ is independent of the other. Then, in consequence of

(3.14) $$C_1 \cap C_2 = \emptyset ,$$

we may write

(3.15) $$C_1 \cup C_2 = \{P_1,\ldots,P_N,Q_1,\ldots,Q_M\}.$$

It follows from the independence of $C_1 \cup C_2$ that both C_1 and C_2 are independent individually. So, in virtue of Theorem 3.3,

(3.16) $$\|C_1\| = |P_1|\ldots|P_N| \quad \text{and} \quad \|C_2\| = |Q_1|\ldots|Q_M| .$$

Thus, by (3,15), (3,7) on the basis of Theorem 3.2, and (3.16),
$$\| C_1 \cup C_2 \| = |P_1|\ldots|P_N|\,|Q_1|\ldots|Q_M| = \|C_1\|\,\|C_2\| .$$

Inversely, suppose that (3.13) holds for the individually independent cores C_1 and C_2. We are going to show (3.14) as well as the independence of $C_1 \cup C_2$.

We prove that the opposite of (3.14) is impossible since $\mathcal{C}_1 \cap \mathcal{C}_2 = \{R_1,\dots,R_m\} \neq \emptyset$ leads to a contradiction. Indeed, in this case, $\mathcal{C}_1$ and $\mathcal{C}_2$ may be written in the form

$$\mathcal{C}_1 = \{P_1,\dots,P_N,R_1,\dots,R_m\},\ \mathcal{C}_2 = \{Q_1,\dots,Q_M,R_1,\dots,R_m\}$$

where N=1 and $P_1=E$ if $\mathcal{C}_1 = \mathcal{C}_1 \cap \mathcal{C}_2$ and M=1 and $Q_1=E$ if $\mathcal{C}_2 = \mathcal{C}_1 \cap \mathcal{C}_2$ / and, naturally, by $R_n \in \mathcal{P}(S)$, we have

(3.17) $$\forall\, n\in\{1,\dots,m\} : |R_n| \geq 2 .$$

Since $\mathcal{C}_1$ and $\mathcal{C}_2$ are independent individually, by Theorem 3.3,

(3.18) $$\|\mathcal{C}_1\| = |P_1|\dots|P_N|\cdot|R_1|\dots|R_m| ,$$

(3.19) $$\|\mathcal{C}_2\| = |Q_1|\dots|Q_M|\cdot|R_1|\dots|R_m| .$$

Now, we show that $\mathcal{C}_1 \cup \mathcal{C}_2$ is independent. Namely, in the contrary case , by Theorem 3.2, (3.18) and (3.19), $\|\mathcal{C}_1 \cup \mathcal{C}_2\| = |P_1\dots P_N R_1\dots R_m Q_1\dots Q_M| \leq |P_1|\dots|P_N||R_1|\dots|R_m||Q_1|\dots|Q_M| = \|\mathcal{C}_1\||Q_1|\dots|Q_M| < \|\mathcal{C}_1\||Q_1|\dots|Q_M||R_1|\dots|R_m| = \|\mathcal{C}_1\|\|\mathcal{C}_2\|$ which contradicts (3.13). Thus, $\mathcal{C}_1 \cup \mathcal{C}_2$ is independent and so, by Theorem 3.3.(ii) and (3.13) ,

$|P_1|\dots|P_N||R_1|\dots|R_m||Q_1|\dots|Q_M| = \|\mathcal{C}_1 \cup \mathcal{C}_2\| = \|\mathcal{C}_1\|\|\mathcal{C}_2\| = |P_1|\dots|P_N||R_1|\dots|R_m||Q_1|\dots|Q_M||R_1|\dots|R_m|$, whence we get $1 = |R_1|\dots|R_m|$ which is impossible by (3.17). Thus (3.14) holds. Then, $\mathcal{C}_1 \cup \mathcal{C}_2$ may be written in form (3.16) where $\mathcal{C}_1=\{P_1,\dots,P_N\}$ and $\mathcal{C}_2=\{Q_1,\dots,Q_M\}$. Thus, by (3.13), we get

$$\|\mathcal{C}_1 \cup \mathcal{C}_2\| = \|\mathcal{C}_1\|\|\mathcal{C}_2\| = |P_1|\dots|P_N||Q_1|\dots|Q_M|$$

whence by Theorem 3.3, the independence of $\mathcal{C}_1 \cup \mathcal{C}_2$ follows. ■

3.3. Core functions

Any function $\phi : \mathcal{C}(S) \to \mathbb{R}$ will be called a core function. In this section, we shall deal with some properties of certain core functions which can come up by the characterization of the capacity of memory.

The size of the core is an important example for a core function. But the size of the core is unsuited for measuring the capacity of memories because the size is not additive. / Any practical measures are additive. We shall define the additivity of core functions later in this section/.

On the other hand, the size of the core is the most essential peculiarity of the core from the point of view of the storage of information. So that we shall say the cores $\mathcal{C}_1$ and $\mathcal{C}_2$ to be equivalent if $\|\mathcal{C}_1\| = \|\mathcal{C}_2\|$.

Previously, in Chapter 1, we gave a characterization of the equivalence relations by means of functions. For a function $f : X \to Y$, the identity of x_1 and x_2 ($x_1, x_2 \in X$) according to f, denoted by $x_1(f)x_2$, was defined /Definition 1.1/ as follows

(3.20) $$x_1(f)x_2 \iff f(x_1) = f(x_2) .$$

We saw that any equivalence relation on X may be written in form (3.20) with an appropriate function f. But, the characterization of equivalence relations by means of functions is not unique. We have proved in Theorem 1.1, that for $f : X \to Y_1$, $g : X \to Y_2$,

$$(f) \leftrightarrow (g) \iff \exists\, \Psi \in BIJ(Y_1, Y_2): g = \Psi(f) ,$$

/where $\leftrightarrow$ denotes the equality of relations and $BIJ(Y_1, Y_2)$ is the set of all bijective functions $\psi: Y_1 \to Y_2$ /.

DEFINITION 3.4. Let the functions $f: X \to Y_1$ and $g: X \to Y_2$ be defined on the non-void set X. If $(f) \leftrightarrow (g)$ then we say that " f is g-like ". /Clearly, if f is g-like then g is f-like/.

REMARK 3.5. Theorem 1.1 may be reformulated in the following way:

THEOREM 1.1/a. For the surjective functions $f: X \to Y_1$, $g: X \to Y_2$, f is g-like iff there exists a bijective function $\lambda: Y_2 \to Y_1$ such that $f = \lambda(g)$.

DEFINITION 3.5. Let f and g be some real functions defined on a set X . The function f is said to be monotonic relative to the function g if $\forall\, x_1, x_2$ 008z***X:

$$g(x_1) < g(x_2) \Longrightarrow f(x_1) < f(x_2) .$$

THEOREM 3.5. Let f,g be some real functions on a set X. If the function f is monotonic relative to g and g is monotonic relative to f then f is g-like.

Proof. First, we show that if f is monotonic relative to g then

(3.21) $\qquad (f) \to (g)$

/ which means that $\forall\, x_1, x_2 \in X: x_1(f)x_2 \Longrightarrow x_1(g)x_2$ /. Indeed, $\forall\, x_1, x_2 \in X: x_1(f)x_2 \iff f(x_1)=f(x_2) \Longrightarrow g(x_1)=g(x_2) \iff x_1(g)x_2$ since the contrary case : $g(x_1)<g(x_2)$ or $g(x_2)<g(x_1)$, yields the false result: $f(x_1) < f(x_2)$ or $f(x_2) < f(x_1)$ in consequence of the monotonity of f relative to g. Since g is monotonic relative to f, we get $(g) \to (f)$ which together with (3.21) result $(f) \leftrightarrow (g)$ i.e. f is g-like. ■

Next, C, C_1, and C_2 denote arbitrary cores of $\mathcal{C}(S)$.

DEFINITION 3.6. A core function $\phi: \mathcal{C}(S) \to \mathbb{R}$ is

positive if $\phi(C) > 0$;

monotonic if $C_1 \subset C_2 \Longrightarrow \phi(C_1) \le \phi(C_2)$;

strongly monotonic if for independent C_2,

$$C_1 \subset C_2 \Longrightarrow \phi(C_1) < \phi(C_2) ;$$

normed if $\|C\| = 2 \Longrightarrow \phi(C) = 1$;

subadditive if $\phi(C_1 \cup C_2) \le \phi(C_1) + \phi(C_2)$;

additive if $\phi(C_1 \cup C_2) = \phi(C_1) + \phi(C_2)$

provided C_1 is independent of C_2 .

LEMMA 3.1. Let $C_2 \in \mathcal{C}(S)$ be independent and $C_1 \in \mathcal{C}(S)$ be an arbitrary subcore of C_2: $C_1 \subset C_2$. Then $\|C_1\| < \|C_2\|$.

Proof. Let $C_2 \in \mathcal{C}(S)$ be independent and C_1 be a subcore of C_2: $C_1 \subset C_2$. Then C_1 is independent also. C_1 and C_2 may be written in the form $C_1=\{P_1,\ldots,P_N\}$, $C_2=\{P_1,\ldots,P_N,Q_1,\ldots,Q_M\}$ and by Theorem 3.3(ii) in consequence of the independence of C_1, we have $\|C_1\| = |P_1|\ldots|P_N|$ and by the independence of C_2,

(3.22) $$\|C_2\| = |P_1|\ldots|P_N|.|Q_1|\ldots|Q_M| = \|C_1\||Q_1|\ldots|Q_M| .$$

Since the partitions $Q_1,\ldots,Q_M$ are proper, we have $|Q_n| \geq 2$ for all $n\in\{1,\ldots,M\}$ and thus, $|Q_1|\ldots|Q_M| \geq 2$ and so from (3.22), we get

(3.23) $$\|C_1\| < \|C_2\| . \quad \blacksquare$$

COROLLARY 3.1. If a core function ϕ is monotonic relative to the size then ϕ is strongly monotonic.

Proof. Take an independent core $C_2 \in \mathcal{C}(S)$ and a subcore C_1: $C_1 \subset C_2$. Then, by Lemmma 3.1, the inequality (3.23) holds. Since ϕ is monotonic relative to the size, it follows by (3.23) that $\phi(C_1) < \phi(C_2)$. $\blacksquare$

Next we give a generalization of Definition 3.3. For the sake of simplicity, we shall suppose from now on that the set S is infinite.

DEFINITION 3.7. (i) A core $C \in \mathcal{C}(S)$ is said to be independent in itself if the memory $\mathcal{M}=(S,C,s)$ is independent in the sense of Definition 2.5 / i.e. every memory cell P of C is independent in C, which means that we can exchange every content of P for any other possible content of P whilst the content of the other memory cells of C remains unchanged /.

(ii) A set $\{C_i \mid i \in \Gamma\} \subset \mathcal{C}(S)$ / where Γ is some set of indexes /, is said to be independent if the union

$$C = \bigcup_{i\in\Gamma} C_i$$

is independent in itself and for all $n,m \in \Gamma$,

(3.24) $$n \neq m \Longrightarrow C_n \cap C_m = \emptyset .$$

REMARK 3.6. Clearly, the set $\{C_1,C_2\}$ of cores is independent in the sense of Definition 3.7 iff C_1 is independent of C_2 in the sense of Definition 3.3.

THEOREM 3.6. If ϕ is an additive core function then

(3.25) $$\phi(C_1 \cup \ldots \cup C_N) = \phi(C_1) + \ldots + \phi(C_N)$$

holds for every finite independent set $\{C_1,\ldots,C_N\} \subset \mathcal{C}(S)$.

Proof. We prove the theorem by induction. For $N = 1$, (3.25) holds trivially. For $N = 2$, (3.25) reduces to the definition of additive core functions. Suppose that (3.25) holds for any independent set of N cores, $/N \geq 2/$. Let $\{C_1,\ldots,C_N,C_{N+1}\} \subseteq \mathcal{C}(S)$ be an arbitrary independent set of $N+1$ cores. Then the set $\{C_1,\ldots,C_N\}$ is independent also. Namely, it is clear that if $n \neq m$ then $C_n \cap C_m = \emptyset$ for all $n,m \in \{1,\ldots,N\} \subset \{1,\ldots,N+1\}$. Moreover, the union $C_0 = C_1 \cup \ldots \cup C_N$ is independent in itself because $C_1 \cup \ldots \cup C_N \cup C_{N+1}$ is independent in itself and C_0 is a subcore of $C_1 \cup \ldots \cup C_{N+1}$. Thus, (3.25) is valid according to the induction's hypothesis. Moreover, C_0 is independent of C_{N+1} /i.e. the set $\{C_0,C_{N+1}\}$ is independent/, since $C_0 \cup C_{N+1}$ is independent /in itself/ and $C_0 \cap C_{N+1} = (C_1 \cap C_{N+1}) \cup \ldots \cup (C_N \cap C_{N+1}) = \emptyset \cup \ldots \cup \emptyset = \emptyset$. Consequently, using the additivity of ϕ by (3.25), we have $\phi(C_1 \cup \ldots \cup C_N \cup C_{N+1}) = \phi(C_0 \cup C_{N+1}) = \phi(C_0) + \phi(C_{N+1}) =$ $= \phi(C_1 \cup \ldots \cup C_N) + \phi(C_{N+1}) = \phi(C_1) + \ldots + \phi(C_N) + \phi(C_{N+1})$. ■

PROPOSITION 3.2. Every positive and additive core function is strongly monotonic.

Proof. Let ϕ be a positive and additive core function and let $C_2 \in \mathcal{C}(S)$ be independent and C_1 be a subcore $C_1 \subset C_2$. Then $C_0 = C_2 \setminus C_1$ is independent of C_1, since $C_0 \cup C_1 = (C_2 \setminus C_1) \cup C_1 = C_2$ is independent in itself and $C_0 \cap C_1 = (C_2 \setminus C_1) \cap C_1 = \emptyset$. Thus by the additivity of ϕ, we have

(3.26) $$\phi(C_2) = \phi(C_0 \cup C_1) = \phi(C_0) + \phi(C_1) ,$$

moreover by the positivity of ϕ, $\phi(C_0) > 0$ and so from (3.26), we obtain $\phi(C_2) = \phi(C_0) + \phi(C_1) > \phi(C_1)$. ■

LEMMA 3.2. For all $C \in \mathcal{C}(S)$ there exists a maximal independent subcore $C_M \subseteq C$.

Proof. If C is independent then the maximal independent subcore of C is $C_M = C$. For the case when C is not independent, in order to find a maximal independent subcore C_M, we give the following simple algorithm.

1 REM in order to initialize the program
it is needed to choose a $P_0 \in C$

2 LET $C_m = \{P_0\}$

3 REM clearly, $C_m = \{P_0\}$ is independent
and $C_m \subset C$. Namely, the case $C_m = C$
is impossible, since C is not independent

4 IF $\forall P \in C \setminus C_m$ ($C_m \cup \{P\}$ is not independent)
THEN LET $C_M = C_m$: STOP

5 REM the program is not stopped since there
exists a $P_1 \in C \setminus C_m$ for which $C_m \cup \{P_1\}$ is
independent

6 LET $C_m = C_m \cup \{P_1\}$: GO TO 4

Since C is not independent, the above program will be stopped at the latest when $|C_m|$ reaches the value $|C| - 1$. ■

3.4. Characterizations of the capacity of memory

If we inquire about the content of a memory $\mathcal{M}$ then we expect a list of content of the individual cells of the core. That is, we expect an information about the content of the vector cell P_0 of the core $\mathbb{C}$ of the memory $\mathcal{M}$. Thus, the capacity of the core $\mathbb{C}$ is not other than the Hartley measure of this information. Namely, according to 2.2.(v) and 2.7.,

$$\text{(3.27)} \qquad \text{cap}(\mathbb{C}) = \log_2 |P_0| \ .$$

Since $|P_0|$ is the size of $\mathbb{C}$, (3.27) may be written in the form

$$\text{(3.28)} \qquad \text{cap}(\mathbb{C}) = \log_2 \|\mathbb{C}\| \ .$$

In the introduction of this chapter, we gave a motivation for (3.28) but only for the case when the size of the core may be written in the form $\|\mathbb{C}\| = 2^m \ / \ m \in \{ 1, 2, \ldots \}$.

One may be pleased with the motivation, that the capacity of the core is not other than Hartley's measure of information about the content of $\mathbb{C}$. But this motivation has a flaw. Namely, there are known many kinds of entropies which are all available for measuring of information. Thus, by the above motivation, it is remained unjustified the privileged role of Hartley's entropy.

In this situation, the problem of characterization of the capacity of memory is an actual question. We know that the size is a good numerical characterization of the capacity of the core. But, the size cannot serve as a meashure of the capacity, since the size is not additive. On the other hand, according to Theorem 3.4, the size is multiplicative. Thus, if $\mathbb{C}_1$ is independent of $\mathbb{C}_2$ then, by (3.28) and (3.13) , we get

$$\text{cap}(\mathbb{C}_1 \cup \mathbb{C}_2) = \log_2 \|\mathbb{C}_1 \cup \mathbb{C}_2\| = \log_2 (\|\mathbb{C}_1\| . \|\mathbb{C}_2\|) =$$

$$= \log_2 \|\mathbb{C}_1\| + \log_2 \|\mathbb{C}_2\| = \text{cap}(\mathbb{C}_1) + \text{cap}(\mathbb{C}_2) \ ,$$

That is, the capacity (3.28) is an additive core function. If $\|\mathbb{C}_1\| < \|\mathbb{C}_2\|$ then $\text{cap}(\mathbb{C}_1) = \log_2 \|\mathbb{C}_1\| < \log_2 \|\mathbb{C}_2\| = \text{cap}(\mathbb{C}_2)$, which shows that the capacity (3.28) is monotonic relative to the size. If $\text{cap}(\mathbb{C}_1) < \text{cap}(\mathbb{C}_2)$ then $\log_2 \|\mathbb{C}_1\| < \log_2 \|\mathbb{C}_2\|$, whence $\|\mathbb{C}_1\| < \|\mathbb{C}_2\|$. So, we see that the size is monotonic relative to the capacity. Clearly, the capacity is normed. In this way, we proved the following theorem:

THEOREM 3.7. If the capacity of the core is defined by (3.28) , then it fulfils the following four properties:

(I) The capacity is an additive core function.
(II) The capacity is monotonic relative to the size.
(III) The size is monotonic relative to the capacity.
(IV) The capacity is a normed core function. ■

The next theorem is the inverse of the above theorem. It may be thought of as a characterization of the capacity of memory.

THEOREM 3.8. Let cap(C) /the capacity of the core/ be defined as an additive, normed core function which is monotonic relative to the size and the size is monotonic relative to the capacity. Then for all $C \in \mathcal{C}(S)$: $cap(C) = \log_2 \|C\|$.

Proof. On the basis of Theorem 3.5, the capacity is a size-like core function, since it is monotonic relative to the size and the size is monotonic relativ to the capacity. So, according to Theorem 1.1/a, there exists a bijective function $\lambda: \mathbb{N}_2 \to \mathbb{R}$ such that $\forall C \in \mathcal{C}(S)$:

$$cap(C) = \lambda(\|C\|) \quad (3.29)$$

We show that $\mathbb{N}_2 = \{2,3,4,\ldots\}$. As the cardinality of certain finite set, the size is an integer which is greater than 1 on account of Theorem 3.3(i). Moreover, since S is infinite, every integer $n \geq 2$ plays a part as a size of some core. For example, if $S = \{s_0, s_1, \ldots, s_{n-1}, s_n, \ldots\}$ then for the unicellular core

$$C_n = \{\{\{s_0\},\{s_1\},\ldots,\{s_{n-1}, s_n, \ldots\}\}\} \quad /n \geq 2/, \quad (3.30)$$

$$\|C_n\| = n \quad (3.31)$$

Now, we shall show that λ is a monotonic increasing function. Let $n,m \in \mathbb{N}_2$ be such that $n < m$. Let C_n be the core (3.30), and likewise, let C_m be of the form (3.30) with m in place of n. Then, $\|C_n\| = n < m = \|C_m\|$, whence $cap(C_n) < cap(C_m)$, since the capacity is monotonic relative to the size. Thus, by (3.29),

$$\lambda(n) = \lambda(\|C_n\|) = cap(C_n) < cap(C_m) = \lambda(\|C_m\|) = \lambda(m).$$

So, we proved that λ is a monotonic increasing function on $\mathbb{N}_2$.

Now, we extend λ to the whole set $\mathbb{N} = \{1, 2, \ldots\}$ of natural numbers so that the extended function λ_1 takes the value 0 on 1: $\lambda_1(1) = 0$ and for $n > 1$, $\lambda_1(n) = \lambda(n)$. λ_1 is monotonic increasing on $\mathbb{N}$. Clearly, it is sufficient to show that $\lambda_1(0) < \lambda_1(n)$ for $n > 1$. Using that λ is increasing on $\mathbb{N}_2$ and the capacity is normed by (3.32), (3.29) for $n > 1$, we obtain

$$\lambda_1(n) = \lambda(n) \geq \lambda(2) = \lambda(\|C_2\|) = cap(C_2) = 1 > 0 \quad \lambda_1(1),$$

where C_2 is a core of size = 2 .

Next, we shall show that λ_1 is a completely additive number theoretical function i. e. $\forall N,M \in \mathbb{N}$:

$$\lambda_1(NM) = \lambda_1(N) + \lambda_1(M) \quad (3.32)$$

We may restrict the proof for the case when $N,M \in \mathbb{N}_2$, since if N=1 or M=1 then (3.32) is fulfilled trivially because of $\lambda_1(1)=0$. Thus, we have to prove $\forall N,M \in \mathbb{N}_2$:

$$\lambda(NM) = \lambda(N) + \lambda(M) \quad (3.33)$$

Let $C_N = \{P\}$ and $C_M = \{Q\}$ be the unicellular cores where $P = \{K_0, K_1, \ldots, K_{N-1}\}$ and $Q = \{L_0, L_1, \ldots, L_{M-1}\}$ and the classes of P are $K_n = \{s_{nM+m} \mid m \in \Gamma_M\}$ if $n \in \Gamma_{N-1}$ where $\Gamma_M = \{0,1,\ldots,M-1\}$, and $K_{N-1} = \{s_{NM-M}, s_{NM-M+1}, \ldots, s_{NM-1}, s_{NM}, s_{NM+1}, \ldots\}$ and the classes of Q are $L_m = \{s_{nM+m} \mid n \in \Gamma_N\}$ if $m \in \Gamma_{M-1}$ and $L_{M-1} = \{s_{M-1}, s_{2M-1}, s_{3M-1}, \ldots, s_{NM-1}, s_{NM}, s_{NM+1}, \ldots\}$. We see that $\forall n,m) \in \Gamma_N \times \Gamma_M$: $s_{nM+m} \in K_n \cap L_m$, and thus,

(3.34) $\|\mathbb{C}_N \cup \mathbb{C}_M\| = \|\{P,Q\}\| = |PQ| = NM = |P||Q| = \|\mathbb{C}_N\|\|\mathbb{C}_M\|$. Consequently by Theorem 3.4, $\mathbb{C}_N$ is independent of $\mathbb{C}_M$. Thus, by the additivity of the capacity and (3.35), (3.29), we get

$$\lambda(NM) = \lambda(\|\mathbb{C}_N\|\|\mathbb{C}_M\|) = \lambda(\|\mathbb{C}_N \cup \mathbb{C}_M\|) = \mathrm{cap}(\mathbb{C}_N \cup \mathbb{C}_M) =$$
$$= \mathrm{cap}(\mathbb{C}_N) + \mathrm{cap}(\mathbb{C}_M) = \lambda(\|\mathbb{C}_N\|) + \lambda(\|\mathbb{C}_M\|) = \lambda(N) + \lambda(M).$$

In this way, we have proved that λ_1 is an increasing, completely additive number theoretical function. Thus, according to a theorem of P. ERDÖS /[3],[1]/, there exists a constant c such that $\forall\, n \in \mathbb{N}$: $\lambda_1(n) = c \log_2 n$. Whence $\forall\, n \in \mathbb{N}_2$: $\lambda(n) = c \log_2 n$. Thus,(3.29) may be written in the form: $\mathrm{cap}(\mathbb{C}) = c \log_2 \|\mathbb{C}\|$. Since the capacity is a normed core function, it follows from $\|\mathbb{C}_2\| = 2$ that $1 = \mathrm{cap}(\mathbb{C}_2) = c \log_2 \|\mathbb{C}_2\| = c \log_2 2 = c$ and so $\mathrm{cap}(\mathbb{C}) = \log_2 \|\mathbb{C}\|$. ■

REFERENCES

[1] J. Aczél, Z. Daróczy, On Measures of Information and Their Characterizations. Academic Press New York (1975).

[2] John J. Donovan, Systems Programming. McGraw-Hill (1972).

[3] P. Erdös, On the Distribution Function of Additive Functions Ann. of Math. [2] 47, 1-20. (1946).

[4] E. Gesztelyi, Szinfelismerö rendszerek matematikai modelljei. Dissertation. Debrecen (1982).

[5] E. Gesztelyi, The application of generalized functions in the color recognition. Обобщенные функции и их применения в Математической Физике, Труды Международной Конференции Москва, 24-28 ноября 1980г. 149-162.

[6] F. Gécseg, M. Steinby, Tree Automata. Akadémiai Kiadó Budapest (1984).

[7] R.V. Hartley, Transmission of Information. Bell System Tech. J. 7. 535-563. (1928).

[8] L. Kalmár, Un Modele Algébrique de Calculatrice Électronique. 3e Congrès de Calcul et de Traitment de L'Information. Toulouse 14-17 mai 1963. Dunod Éditeur, Paris.

[9] А.Н. Колмогоров, Три Подхода к Определению Понятия "Количество Информации". Проблемы Передачи Информации. Том 1. Вып. 1. 3-11. (1965).

[10] G. Luecke, J.P. Mize, W.N. Carr, Semiconductor Memory Design and Application, McGraw-Hill Book Company (1980).

[11] Gy. Maksa, Információmértékek elöállitása és jellemzése függvényegyenletek segitségével. Dissertation. Debrecen - Miskolc 1985.

[12] L.A. Zadeh, E. Polak, System Theory. McGraw-Hill (1969).

A mathematical theory of processors, I.

E. Geszthelyi

Summary

In the paper, following a new approach of the notion of the memory, a mathematical theory of processors is developed. According to the definition in the paper, a memory on a set S is nothing else than a three-tuple M = (S,C,s), where S is the set of states of the memory M,C -the core of M- is a family of partitions of S and s - the instant state of M- is a variable on S. Every partition $P \in C$ is called a memory cell of M. Let the classes of P be indexed in some way with a low segment of non-negative integers. Every instant state s is contained in exactly one class $K_n \in P : s \in K_n$. It is said in this case that the content of the memory cell P is n at the instant state s. /Thus, the content of a memory cell depends on the instant state/. If a memory on the state set of an automation is created, then a processor comes into being which operates on the content of the memory, in accordance with the classical determination of the processor /[2]/.

A processzorok matematikai elmélete I.

Geszthelyi Ernő

Összefoglaló

A cikkben a szerző a processzorok matematikai elméletét dolgozza ki, amely a memória fogalmának ujfajta felfogásán alapszik. A cikkben leirtak szerint az S halmazon vett memória semmi más mint egy M = (S,C,s) hármas, ahol S az M memória állapotainak halmaza, C - az M magja - az S particióinak egy családja, és s - az M pillanatnyi állapota - egy változó az S-en. Minden P∈C partició az M memória egy cellája. Legyenek a P-k indexelve a nem-negativ számok alsó szegmenseivel. Minden s pontosan egy $K_n \in P$ osztályba tartozik, $K_n \in P : s \in K_n$. Ekkor aut mondjuk, hogy a P tartalma az s állapotban n. /Tehát a memória cella tartalma az s pillanatnyi állapottól függ./ Ha egy automata állapot-halmazán egy memória létesül, akkor ezzel egy processzor is létrejött, amely a memória tartalmán "operál", a processzorok klasszikus meghatározásának megfelelően /[2]/.

MTA SZTAKI Közlemények 37/1987 pp. 63-96

SCALING OF RANDOM VARIABLES AND ARRANGEMENT PROBLEMS IN LAYOUT DESIGN

B. GOETZE and *W. NEHRLICH*

Dept. of Appl. Discrete Math.
Karl-Weierstrass-Inst. of Math.
Acad. Sci. of the GDR
Mohrenstr. 39, Berlin, DDR-1086

1 Introduction

A key problem in Design Automation is the arrangement of components of a large and complicated system. As an important example consider the placement problem, where the modules of a circuit have to be assigned to favourable locations on a placement media. At the strongly idealized level of topological design large systems can be represented by graph- or hypergraph models. Moreover, in many cases it is also adequate to represent the one-, two- or multidimensional placement media by special graphs. (An important example is a grid graph.) Using such models placement problems can be considered as discrete optimization problems, in particular as embedding problems for graphs or hypergraphs. Some examples of embedding problems of this kind are the following NP-hard problems (see /GaJo/, /Jo/):

OPTIMAL LINEAR ARRANGEMENT OF GRAPHS,
MIN CUT LINEAR ARRANGEMENT OF GRAPHS,
MINIMUM BANDWITH, CROSSING NUMBER,
EDGE-EMBEDDING ON A GRID,
MINIMUM AREA EMBEDDING OF PLANAR GRAPHS,
WEIGHTED GRAPH EMBEDDABILITY.

Stimulated by the practical importance of such problems many attempts and different approaches have been made in order to find approximative solutions or solutions for subproblems. Among the numerous heuristic solution procedures for the various kinds of problems one can find greedy strategies, iterative and Monte-Carlo-procedures and even such kinds of solution

methods which are founded on non-discrete mathematical methods. Different from those discrete strategies which proceed stepwise on the basis of local decision, the non-discrete models represent global optimization criteria. Examples of non-discrete approaches and strategies concerning one- and two-dimensional arrangement problems can be found in /QuiBr/, /Otten1/, /Otten2/, /Fuku1/, /May/. In our paper we deal with a non-discrete heuristic approach which is based on a model of mathematical statistics - the scaling of random variables by optimization of the correlation coefficient.

Starting from an idea of /Fuku/ we shall apply this model to various arrangement problems, where the two-dimensional embedding of hypergraphs is in the centre of our attention. The mathematical treatment of this approach is explained in detail. Furthermore, the embedding algorithm is generalized to the case of additional placement constraints by incorporating these constraints into the model from the beginning. A natural and important application of this approach is the layout design for electronic circuits.

2 Scaling by Optimization of Correlation Coefficient

Here we consider the following problem: Let two discrete random variables be given, then we search for a scaling of these random variables such that the correlation coefficient is maximum. This problem leads to the determination of extremal points of quadratic forms, which is treated in detail in standard literature (cf. /Ga/, /Co/). In subsequent chapters we will apply scaling theory to the solution of certain arrangement problems. Though the mathematical solution of the scaling problem appears in full detail in statistics literature, we shall present a short derivation of this solution here, so that we can refer to this derivation in § 5.

Let two random variables $\tilde{x}$ and $\tilde{y}$ be given, which are varying within the sets $\tilde{X} = \{\tilde{x}_1, \ldots, \tilde{x}_m\}$ and $\tilde{Y} = \{y_1, \ldots, y_n\}$, respectively. Here $\tilde{x}_i$ and $\tilde{y}_j$ are arbitrary elementary events. Let the probability distribution on $\tilde{X} \times \tilde{Y}$ be given by the matrix $P = (p_{ij})_{m,n}$. This means that p_{ij} is the probability of the event "$\tilde{x} = \tilde{x}_i$ and $\tilde{y} = \tilde{y}_j$". For short, we will write $rp_i = \sum_{j=1}^{n} p_{ij}$ and $cp_j = \sum_{i=1}^{m} p_{ij}$ for row- and column-sums, respectively. These values represent the probability of "$\tilde{x} = \tilde{x}_i$" and "$\tilde{y} = \tilde{y}_j$", respectively.

If we are scaling the $\tilde{x}_i$ and $\tilde{y}_j$ to positions on the x-axis and the y-axis, then from $\tilde{x}$ and $\tilde{y}$ we obtain real random variables x and y, respectively. This permits to define the expectation values:

$$Ex = \sum_{i=1}^{m} rp_i \cdot x_i \quad , \quad Ex^2 = \sum_{i=1}^{m} rp_i \cdot x_i^2 \quad \text{(the definition for y is analogous)},$$

$$E(x \cdot y) = \sum_{i=1}^{m} \sum_{j=1}^{n} x_i \cdot p_{ij} \cdot y_j \ .$$

From this definition we may also define the variances and the covariance:

$\mathrm{Var}(x) = Ex^2 - (Ex)^2$ (analogous for y) ,

$\mathrm{Cov}(x,y) = E(x \cdot y) - Ex \cdot Ey$.

The correlation coefficient is defined as follows:

$$\varrho_{x,y} = \frac{\mathrm{Cov}(x,y)}{(\mathrm{Var}(x) \cdot \mathrm{Var}(y))^{1/2}}$$

The correlation coefficient ranges in size between -1 and +1 . If $|\varrho_{x,y}| = 1$, then the variables x and y are deterministically correlated, and the (x_i, y_j) with $p_{ij} \neq 0$ are all located on a straight line. If $\varrho = +1$, then this straight line has a positive slope, for $\varrho = -1$ the slope is negative.

The value of $|\varrho_{x,y}|$ is a measure for the concentration of the probability density function P around a straight line. Notice, that the correlation coefficient has the property of being invariant with respect to linear transformations, i.e.

$$\varrho_{(ax+b,cy+d)} = \varrho_{x,y} \ .$$

Our aim is a scaling of the random variables $\tilde{x}$ and $\tilde{y}$ (i.e. to find values x_i and y_j) such that we obtain a maximum value of ϱ .

First, we can assume without loss of generality, that the matrix P possesses neither zero-rows nor zero-columns. This is because an elementary event with zero-probability could be scaled on any place without influencing the value ϱ . Since ϱ is invariant with respect to linear transformations, we can assume the values of x and y to be normalized such that

$$Ex = Ey = 0 \qquad (2.1.)$$

$$\mathrm{Var}(x) = \mathrm{Var}(y) = 1 \ . \qquad (2.2.)$$

From (2.1.) we obtain

$\mathrm{Var}(x) = Ex^2$, $\mathrm{Var}(y) = Ey^2$ and $\mathrm{Cov}(x,y) = E(x \cdot y)$.

Thus, from (2.1.) and (2.2.) we conclude the simplified formula

$$\varrho_{x,y} = E(x \cdot y) \ . \qquad (2.3.)$$

We use the following notations:

$$x = (x_1,\ldots,x_m)^T \ , \ y = (y_1,\ldots,y_n)^T, \ e_m = (\underbrace{1,\ldots,1}_{m})^T \ ,$$

and

$$R_P = \begin{pmatrix} rp_1 & & 0 \\ & \ddots & \\ 0 & & rp_m \end{pmatrix} , \quad C_P = \begin{pmatrix} cp_1 & & 0 \\ & \ddots & \\ 0 & & cp_n \end{pmatrix} .$$

Thus, we can write

$$Ex = x^T \cdot R_P \cdot e_m \ , \quad Ey = y^T \cdot C_P \cdot e_n \ ,$$

$$Ex^2 = x^T \cdot R_P \cdot x \quad \text{and} \quad Ey^2 = y^T \cdot C_P \cdot y \ .$$

The normalization conditions (2.1.) and (2.2.) then read as follows:

$$x^T \cdot R_P \cdot e_m = y^T \cdot C_P \cdot e_n = 0 \qquad (2.1.a)$$

$$x^T \cdot R_P \cdot x = y^T \cdot C_P \cdot y = 1 \ . \qquad (2.2.a)$$

Finally, expression (2.3.) has the form

$$\varrho = x^T \cdot P \cdot y \ . \qquad (2.3.a)$$

Now, the problem is to determine extremal points of $\varrho = x^T \cdot P \cdot y$ under the conditions (2.1.a) and (2.2.a). By the use of Lagrangian multipliers we obtain necessary conditions for local extremal points under certain conditions.

First we will consider only the second restriction from (2.2.a), i.e.

$$y^T \cdot C_P \cdot y = 1 \ .$$

Thus we have to define a function

$$H(x_1,\ldots,x_m,y_1,\ldots,y_n) = x^T \cdot P \cdot y - \varrho' \cdot y^T \cdot C_P \cdot y \ .$$

Necessary conditions for extremal points are:

$$\frac{\partial H}{\partial y_j} = 0 \qquad \text{for} \quad j = 1,\ldots,n$$

or, for short,

$$\frac{\partial H}{\partial y} = 0 \ .$$

Hence we obtain

$$\frac{\partial}{\partial y} (x^T \cdot P \cdot y) = \varrho' \cdot \frac{\partial}{\partial y} (y^T \cdot C_P \cdot y)$$

$$P^T \cdot x = 2 \varrho' \cdot C_P \cdot y \ . \qquad (2.4)$$

In order to clarify the meaning of the factor $2\varrho'$ we multiply equation (2.4.) by y^T and obtain

$$y^T \cdot P^T \cdot x = 2\varrho' \cdot y^T \cdot C_P \cdot y \ .$$

Under the above condition this leads to

$$x^T \cdot P \cdot y = 2\varrho' \quad , \text{ i.e.}$$

$$\varrho = 2\varrho' \ .$$

Thus, from (2.4.) we obtain

$$P^T \cdot x = \varrho \cdot C_P \cdot y \qquad (2.4.a)$$

Since the matrix P does not contain any zero-rows or zero-columns, R_P and C_P are regular matrices.

Thus, from (2.4.a) we obtain

$$\varrho \cdot y = C_P^{-1} \cdot P^T \cdot x \ . \qquad (2.4.b)$$

On the other hand, multiplying (2.3.a) by ϱ , we have

$$\varrho^2 = \varrho \cdot x^T \cdot P \cdot y = x^T \cdot P \cdot (\varrho \cdot y) \ .$$

Substituting (2.4.b) we conclude

$$\varrho^2 = x^T \cdot (P \cdot C_P^{-1} \cdot P^T) \cdot x \ . \qquad (2.5.)$$

Equation (2.5.) describes ϱ^2 as a quadratic form in x . Our aim is to find maximum values of ϱ^2 under the conditions (2.1.a) and (2.2.a). Thus, we seek for

$$\max_{\substack{x^T \cdot R_P \cdot x = 1 \\ x^T \cdot R_P \cdot e_m = 0}} x^T \cdot (P \cdot C_P^{-1} \cdot P^T) \cdot x \qquad (2.6.)$$

For this purpose, the well-known theorem about extremal points of quadratic forms (cf. /Co/, /Ga/) can be used.

Theorem

Let S and D be $(m \times m)$-matrices, S being symmetric and D positive definite. Let the (real) eigenvalues of

$$S \cdot x = \lambda \cdot D \cdot x$$

be $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_m$, and let $x^{(i)}$ be the eigenvector corresponding to λ_i. Then

$\lambda_i = \max \{ x^T \cdot S \cdot x \mid x^T \cdot D \cdot x = 1$ and $x^T \cdot D \cdot x^{(j)} = 0$ for all $j = 1, \ldots, i-1 \}$,
where the maximum value is assumed for $x = x^{(i)}$.

Hence, for the special case $i = 2$ the theorem yields

$$\lambda_2 = \max_{\substack{x^T \cdot D \cdot x = 1 \\ x^T \cdot D \cdot x^{(1)} = 0}} x^T \cdot S \cdot x$$

where the maximum value is assumed for $x = x^{(2)}$. For our purpose the theorem can be applied to (2.6.) with $S = (P \cdot C_P^{-1} \cdot P^T)$ and $D = R_P$. The maximum value (2.6.) equals to the second-largest eigenvalue of

$$(P \cdot C_P^{-1} \cdot P^T) \cdot x = \lambda \cdot R_P \cdot x \quad \text{or}$$

$$\boxed{(R_P^{-1} \cdot P \cdot C_P^{-1} \cdot P^T) \cdot x = \lambda \cdot x} \qquad (2.7.)$$

For the proof of this assumption it remains to show, that $x^{(1)} = e_m$ holds.

Lemma

The matrix $R_P^{-1} \cdot P \cdot C_P^{-1} \cdot P^T$ is a stochastic matrix.

Proof. Both the matrices $N_1 = R_P^{-1} \cdot P$ and $N_2 = C_P^{-1} \cdot P^T$ are clearly stochastic, i.e. all items are nonnegative and all row sums equal to one. Thus, $N_1 \cdot N_2$ is again a stochastic matrix.

According to the lemma, (2.7.) has the largest eigenvalue $\lambda_1 = 1$, with e_m as a corresponding eigenvector.

We discuss some more conclusions from the theorem.

Notice, that $Ex^{(2)} = x^{(2)T} \cdot R_P \cdot e_m = x^{(2)T} \cdot R_P \cdot x^{(1)} = 0$.
The same holds true for the other eigenvectors $x^{(3)}, \ldots, x^{(m)}$.
The eigenvalues $\lambda_3, \lambda_4, \ldots$ are also extremal points of φ^2 under the conditions (2.1.a) and (2.2.a), where the additional condition of orthogonality of the vectors (with respect to the matrix R_P) holds.

Hence it is evident, that all eigenvalues of (2.7.) are real and positive: $1 = \lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_m \geqslant 0$. Notice that these properties can also be seen directly from the structure of (2.7.). First note that (2.7.) is equivalent to

$$(R_P^{-1/2} \cdot P \cdot C_P^{-1} \cdot P^T \cdot R_P^{-1/2}) \cdot z = \lambda \cdot z \quad ,$$

which is a symmetric problem, and therefore all eigenvalues are real ones. Furthermore, let $F = R_P^{-1/2} \cdot P \cdot C_P^{-1/2}$, then the problem has the form

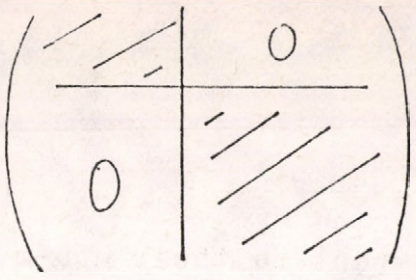
$$(F \cdot F^T) \cdot z = \lambda \cdot z \quad .$$

Hence $\lambda \geqslant 0$ follows from the fact, that this matrix is positive definite.
Now we consider a further fact concerning the spectrum of (2.7.)

Lemma

For the eigenvalue problem (2.7.) the property $1 > \lambda_2$ holds if and only if the matrix P is connected.

A matrix P is called unconnected iff there are permutations of the rows and columns transforming P into



Given a $(m \times n)$ matrix P we can define a corresponding graph $G_P = (V,E)$ in a canonical way as follows:

$V=\{1,\ldots,m\}$ and $(i,k) \in E \leftrightarrow \exists j \in \{1,\ldots,n\}: p_{ij} \neq 0 \wedge p_{kj} \neq 0$.

The graph defined in this way is connected (in the usual sense) iff the matrix P is connected.

Proof of the lemma

Let G_P be the canonical graph corresponding to P. Consider the matrix $B = (R_P^{-1} \cdot P \cdot C_P^{-1} \cdot P^T)$ of the eigenvalue problem (2.7.). Clearly, sgn (B) is the adjacency matrix of G_P. Then P is connected iff sgn (B) is connected. Now the theorem of PERRON/FROBENIUS (cf./Ga/) can be applied to the stochastic matrix B. This yields the assumption of the lemma.

The claim of P being connected will be natural for our applications (cf. subsequent chapters). Thus, for the spectrum of (2.7.) we can presume

$$1 = \lambda_1 > \lambda_2 \geqslant \lambda_3 \geqslant \ldots \geqslant \lambda_n \geqslant 0 .$$

Hence, the eigenvalue λ_2 together with the vector $x = x^{(2)}$ is a nontrivial solution for the problem (2.6.).

Similarly to (2.7.), one can derive an eigenvalue problem for the vector y:

$$(C_P^{-1} \cdot P^T \cdot R_P^{-1} \cdot P) \cdot y = \lambda \cdot y \qquad (2.8.)$$

Again, for the eigenvalues the equation $\lambda = \varrho^2$ holds. If we take

$$N_1 = R_P^{-1} \cdot P \quad \text{and} \quad N_2 = C_P^{-1} \cdot P^T \ , \qquad (2.9.)$$

then the problems (2.7.) and (2.8.) have the form

$$(N_1 \cdot N_2) \cdot x = \lambda \cdot x \quad \text{and}$$
$$(N_2 \cdot N_1) \cdot y = \lambda \cdot y \quad .$$

Such problems are equivalent with respect to their non-zero eigenvalues (including the multiplicities). The relations between equivalent eigenvectors are given by

$$x = N_1 \cdot y \quad \text{and} \quad y = N_2 \cdot x \quad .$$

In our case we obtain the following formulas:

$$x = (R_P^{-1} \cdot P) \cdot y \quad \text{and}$$
$$y = (C_P^{-1} \cdot P^T) \cdot x \quad . \qquad (2.10.)$$

3 Bandshape Optimization for Rectangular Matrices

There are numerous practical applications, where bandshape optimization for rectangular matrices is a key problem. Bandshape optimization means to permute rows and columns of a given matrix in such a way that the non-zero elements (and especially the "heavy" elements) are located as tight as possible around a diagonal line.

Example

$$A = \begin{pmatrix} 1&0&0&0&5 \\ 0&0&1&5&0 \\ 1&1&0&0&0 \\ 0&1&0&1&0 \\ 0&1&0&0&0 \\ 0&0&6&0&0 \\ 6&0&0&0&1 \end{pmatrix} \qquad A' = \begin{pmatrix} 6&0&0&0&0 \\ 1&5&0&0&0 \\ 0&1&1&0&0 \\ 0&0&1&0&0 \\ 0&0&1&1&0 \\ 0&0&0&6&1 \\ 0&0&0&1&5 \end{pmatrix}$$

If the row-sequence of A is permuted according to $\pi_0 = (6,2,4,5,3,7,1)$, and the column-sequence of A is permuted according to $\sigma_0 = (3,4,2,1,5)$, then we obtain the matrix A' which possesses a very good bandshape.

The fuzzy notion of bandshape can be made precise with the help of the correlation coefficient, regarding the given matrix as a probability density function.

Let $A = (a_{ij})_{m,n}$ be a real nonnegative rectangular matrix with $A \neq 0$. Then we turn to the the matrix

$$P = \frac{1}{a} \cdot A \quad , \quad \text{where} \quad a = \sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij} \, . \qquad (3.1.)$$

Then P is a probability distribution on $\{x_1,\ldots,x_m\} \times \{y_1,\ldots,y_n\}$. The correlation coefficient ρ for the distribution P under the special assignment $x_i = i$ and $y_j = j$ represents one possible measure of the bandshape of A (or P respectively). The closer ρ is to +1 , the better is the bandshape of A. For our example, we obtain $P = \frac{1}{30} \cdot A$ and $\rho \approx -0.5931$. The permutation $\pi_0 = (6,2,4,5,3,7,1)$ corresponds to the new assignment $x_1 = 7$, $x_2 = 2$, $x_3 = 5$, $x_4 = 3$, $x_5 = 4$, $x_6 = 1$, $x_7 = 6$.

The permutation $\sigma_o = (3,4,2,1,5)$ corresponds to $y_1=4$, $y_2=3$, $y_3=1$, $y_4=2$, $y_5=5$. For the new assignment we obtain a correlation coefficient $\varrho' \approx +0.9712$, which expresses the bandshape quality of A'.

Let us now assume the given matrix A to be connected. Hence, the matrix A does not have any zero-rows or -columns. Clearly, P is also connected. The re-arrangement of rows and columns of A can be determined by scaling of the variables x_i and y_j. The eigenvalue-problem (2.7.) can be reformulated in the following form:

$$\boxed{(R^{-1} \cdot A \cdot C^{-1} \cdot A^T) \cdot x = \lambda \cdot x} \quad , \qquad (3.2.)$$

where $R = R_A$ and $C = C_A$. Then the eigenvector corresponding to the largest nontrivial eigenvalue λ_2 of (3.2.) yields the optimal real values for the x_i. According to (2.4.b) or (2.10.) the corresponding scaling of the y_j is derived from the transformation

$$y = (C^{-1} \cdot A^T) \cdot x \quad .$$

Notice, that x and y are of course real vectors and thus in general do not represent permutations of the rows and columns of A. The real components of these vectors have to be transformed to discrete ones. This can be performed simply by sorting the components.

For our example, the largest nontrivial solution is

$\lambda \approx 0.9671$ with

$$x = \begin{pmatrix} 0.0888 \\ -0.0807 \\ 0.0411 \\ -0.0368 \\ 0.0022 \\ -0.1048 \\ 0.0815 \end{pmatrix} \qquad y = \begin{pmatrix} 0.0773 \\ 0.0022 \\ -0.1014 \\ -0.0734 \\ 0.0875 \end{pmatrix}$$

The sorting of the components leads also to the permutations π_o and σ_o, which produce the above matrix A'.

The correlation coefficient is $\varrho = \sqrt{\lambda} \approx 0.9834$, whereas the matrix A' has only a correlation coefficient $\varrho' \approx 0.9712$. The loss of bandshape quality is connected with the transformation of the optimal eigensolution to the (discrete) sorted components. This is the reason why the method described here does not necessarily produce an optimal permutation of the given matrix. It is an open problem to evaluate the quality of approximation of this non-discrete method.

The method described here was presented in various forms and independently in /Fuku1/, /Otten1/ and /May/ (see also /Fuku2/, /MaMe/). Note that the derivation of this heuristic method with the help of the correlation coefficient - as described above - is due to /Fuku1/. In /Otten1/ the problem MIN CUT LINEAR ARRANGEMENT of hypergraphs was considered.[1] This problem reads as follows:

Let a finite hypergraph $H = (V,E)$ be given. Determine an embedding (i.e. a one-one-mapping) $\pi : V \to \{1,2,\ldots,|V|\}$ such that the out-width (or track number)

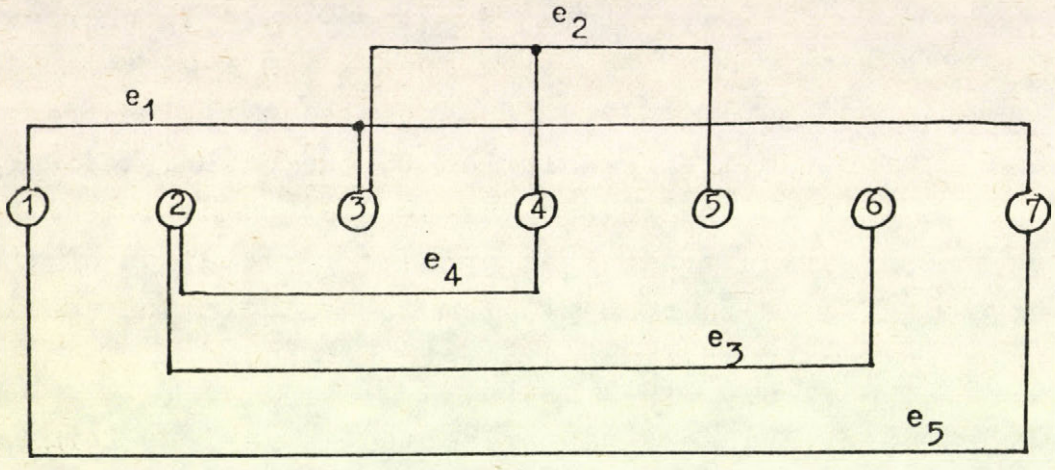
$$\max_{1 \le i \le |V|-1} \left| \{ e : e \in E \text{ and } \min_{u \in e} \pi(u) \le i < \max_{v \in e} \pi(v) \} \right| \text{ is minimum.}$$

This problem is NP-hard. (In 1974 STOCKMEYER proved that the MINCUT LINEAR ARRANGEMENT problem is NP-hard for graphs, cf. /GaJo/. This problem is NP-hard even when restricted to graphs with degree ≤ 3, whereas it becomes solvable in time $O(n \log n)$ for arbitrary trees, see /CMST/.)

As an example, consider the hypergraph $H = (\{1,\ldots,7\}, \{e_1,\ldots,e_5\})$, where $e_1 = \{1,3,7\}$, $e_2 = \{3,4,5\}$, $e_3 = \{2,6\}$, $e_4 = \{2,4\}$, $e_5 = \{1,7\}$ together with the embedding $\pi(i) = i$. We have the following figure:

---

[1] This problem is often called Board Permutation problem.



The cut-width for $\pi(i)=i$ equals 5. It is assumed for $i=3$.

Let us consider the incidence matrix of the hypergraph.

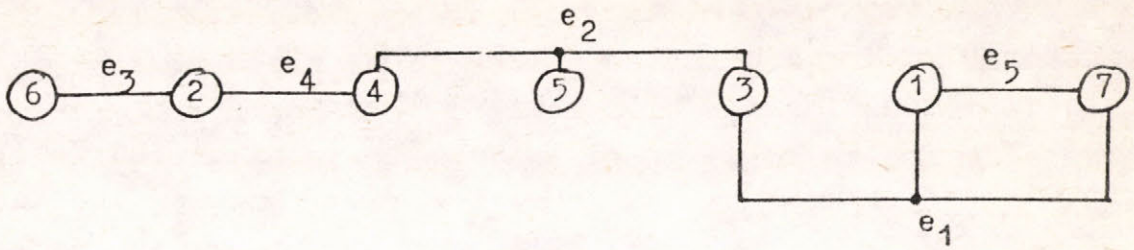
$$A = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

Here the rows correspond to the vertices, and the columns correspond to the edges of H.

The lines in the above figure (corresponding to the edges) appear as non-zero intervals, i.e. the regions between the first and the last non-zero element in the columns of A. Thus one can expect a reduction of the cut-width by reducing the lengths of these intervals. This goal clearly corresponds to the bandshape optimization for the incidence matrix. For our example, we obtain the above mentioned permutations π_0 and σ_0 for the rows and columns, respectively. The resulting matrix is

$$A' = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

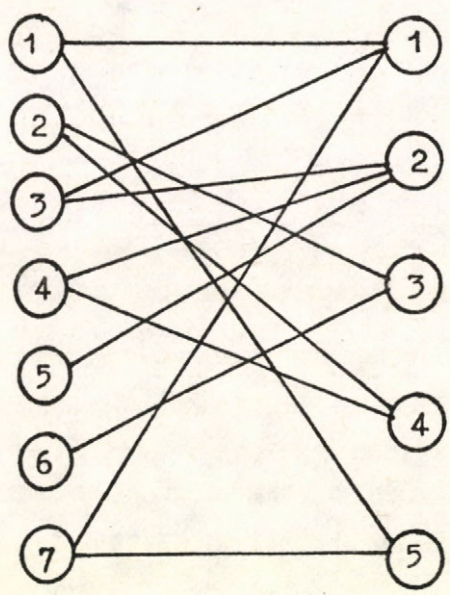
The corresponding re-arrangement of the hypergraph yields



The cut-width under this embedding equals 2.

Another arrangement problem is considered in /May/, /MaMe/ where bipartite graphs are investigated.

For an illustration of this problem consider the following example.

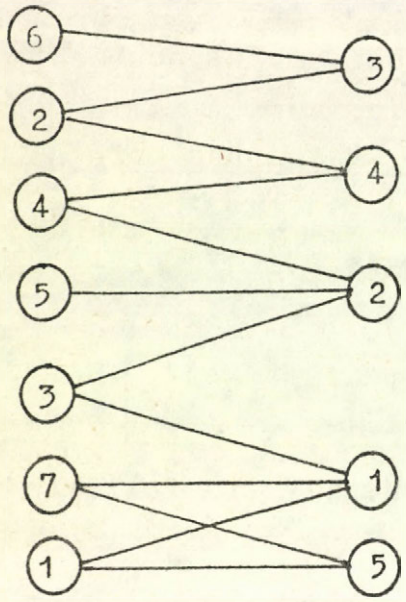


Roughly speaking, the problem consists of finding permutations of both columns such that we obtain a clear and readable representation of the graph. This fuzzy description was made precize in /May/ by formulating the following tasks:

a) Minimize the total sum of edge-lengths.
b) Minimize the number of edge crossings.

The heuristic procedure developed in /May/ is based on the algorithm described above.
This can be seen by describing the given bipartite graph by a rectangular 0/1-matrix, whose rows/columns correspond to the "left" / "right" vertices of the graph. For our example this matrix coincides with the incidence matrix A of the hypergraph presented above. (This is because hypergraph and bipartite graph are equivalent notions if we identify "right" vertices with edges and "left" vertices with the vertices of the hypergraph.) Thus we can expect, that a bandshape optimization will reduce both the total sum and the crossing number

for bipartite graphs. For our example, the permutations π_o and σ_o mentioned above produce the following representation of the bipartite graph.



Notice, that - similarly to the MINCUT problem - no results concerning the quality of the approximation are known for this optimization problem.

Now we shall discuss another aspect of the eigenvalue problems (2.7.) and (3.2.). It is connected with the power method (v. Mises method) which can be applied here.

Let $B \cdot x = \lambda \cdot x$ be an eigenvalue problem, and assume that all eigenvalues are real and nonnegative. The power method consists of the iteration of $x := B \cdot x$, where the vector x has to be normalized from time to time during the iteration process. In our problem we have

$B = N_1 \cdot N_2$, where

$N_1 = R^{-1} \cdot A$ and $N_2 = C^{-1} \cdot A^T$.

Since N_1 and N_2 represent the transformations between x and y (cf. 2.10.), we can divide the iteration step of the power method into:

$$y := N_2 \cdot x \qquad (3.3.a)$$

$$x := N_1 \cdot y \qquad (3.3.b)$$

Consider (3.3.a) :

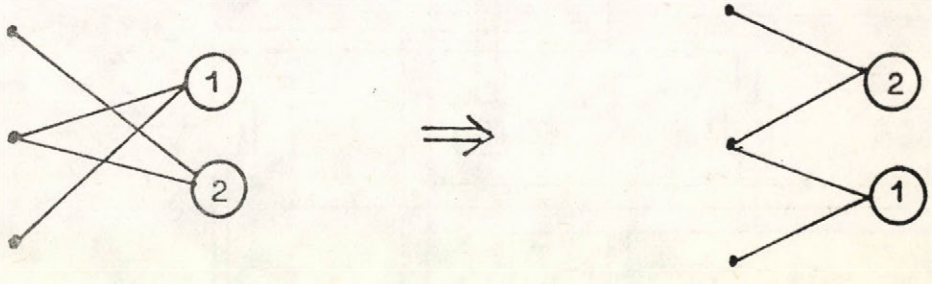
$y := C^{-1} \cdot A^T \cdot x$

$$y_j := \frac{1}{ca_j} \cdot \sum_{i=1}^{m} a_{ij} \cdot x_i \qquad j=1,\ldots,n \; . \qquad (3.4.a)$$

Analogously, from (3.3.b) we get:

$$x_i := \frac{1}{ra_i} \cdot \sum_{j=1}^{n} a_{ij} \cdot y_j \qquad i=1,\ldots,m \; . \qquad (3.4.b)$$

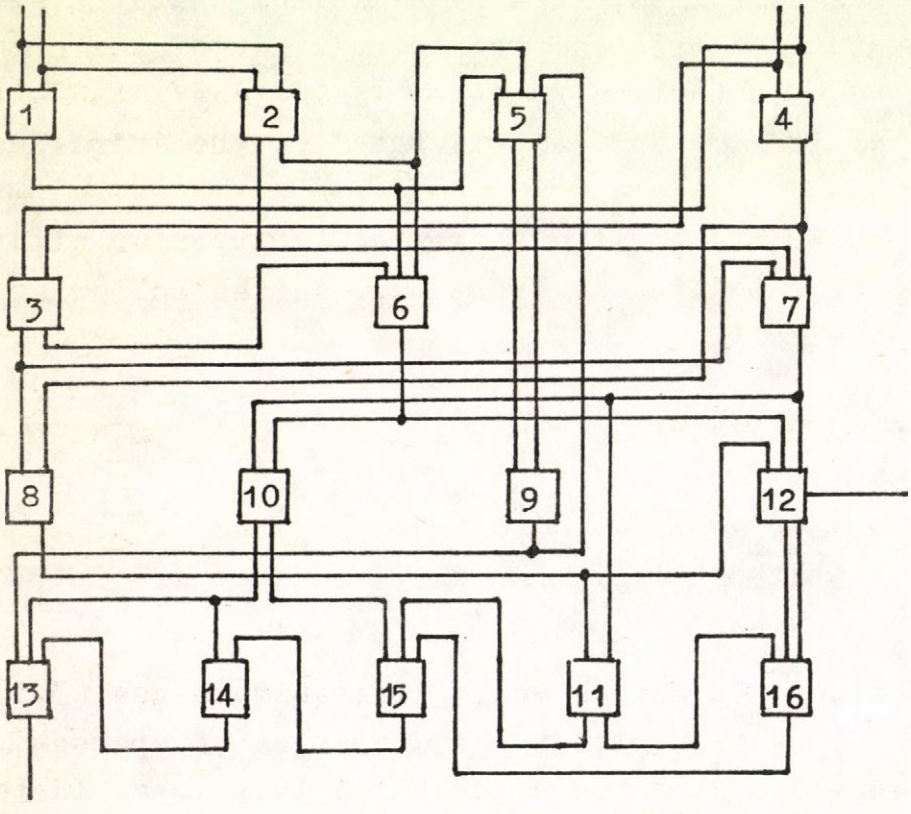
Now we turn back to the arrangement problem for bipartite graphs. Let x_i be the (real) locations of the m "left" vertices. Then applying step (3.4.a) means, that the "right" vertices are shifted into the center of gravity of its "left" neighbours. Similarly, applying (3.4.b) means to shift the "left" vertices into the center of gravity of its "right" neighbours. Therefore, the successive iteration process, alternating (3.4.a) and (3.4.b), is called "averaging". The averaging process converges, if an equilibrium state is attained. Those states correspond to the solution of the eigenvalue problem. Notice that in /Otten1/ and /May/ the heuristic procedures are derived and based on the averaging model.
Note that the averaging positions are really superior with respect to the minimization of crossings, as the following example demonstrates.



Finally, notice that it may be very profitable to use (3.3.a) and (3.3.b) in connection with the application of sparse-matrix techniques, even in the case that A is sparse. Instead of $B = N_1 \cdot N_2 = R^{-1} \cdot A \cdot C^{-1} \cdot A^T$, (which may be very dense in comparison with A) one has only to handle the sparse matrix A and the diagonal matrices R^{-1} and C^{-1} .

4 Placement of Circuits in the Plane

Design-automation for electronic circuits includes searching for sophisticated methods for the solution of placement and routing problems on the level of topological design. This level is characterized by a more or less rigorous idealization inasmuch as the geometrical shape and size of the modules and of the connecting wires are disregarded. Thus, in topological design, usually the representation of circuits by hypergraphs is a sufficient model. The modules correspond to the vertices, and the nets (signal sets) are represented by the edges of the hypergraph. Consider the following example of an electronic circuit.



The corresponding hypergraph has 16 vertices and 25 edges. The following table defines an enumeration $e_1,\ldots,e_{25}$ of the edges.

The table shows the incidence matrix of the given hypergraph.

i \ j	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25
1	1	1		1																					
2	1	1	1		1																				
3						1	1	1	1																
4							1	1		1															
5			1	1							1	1											1		
6			1	1		1							1												
7					1				1	1				1											
8									1	1					1										
9											1	1											1		
10													1	1		1						1			
11														1	1					1	1				
12													1	1	1									1	1
13																1	1						1		
14																1	1	1							
15																		1	1	1		1			
16																			1		1			1	1

Notice that the model of an edge-weighted hypergraph may be more efficient in practical computation. In our example, we could join the edges e_1 and e_2, e_7 and e_8, e_{11} and e_{12}, e_{24} and e_{25}, respectively, to constitute new edges with weight = 2. For the subsequent discussion, however, it is more convenient to handle the simpler model of unweighted hypergraphs.

We consider the topological placement problem, i.e. the problem to locate the modules (vertices) in the plane in such a way that the subsequent routing process is able to find a "good" wiring. We shall assume that a wiring is good if it has minimum (or at least near-optimal) total wire-length. Since - in the placement step - the real final wiring has not yet been performed, the total wire-length has to be approximated in a certain way. In spite of the concrete model used for this estimation, a general goal for any placement procedure should be that strongly connected modules are located close together, i.e. that proximity reflects connectivity.

Notice that the connectivity of vertices intuitively determines distances d_{ij} between vertices. The distances derived from connectivity (and possibly from additional data) may be defined in various ways (cf. /Fuku/, /Otten2/) . In any case, a good placement result will be an embedding of the given hypergraph in Euclidean plane such that the distances are preserved "as well as possible".[1] In mathematical statistics, this problem is called the multidimensional scaling problem (MDS). The formulation of the placement problem as MDS-problem is an idea due to OTTEN, and it was pointed out in /Otten1/ and /Otten2/. However, we shall not use here the classical MDS-solution (due to SCHOENBERG, cf. /MaKeBi/) which was applied in /Otten2/ . We consider a slightly different solution, proposed in /Fuku/. The investigation of this approach leads to a generalization which will be presented in § 5.

The problem of "scaling of a hypergraph" can be treated by solving the eigenvalue problem (3.2.), maximizing the correlation coefficient. The eigenvector corresponding to the largest nontrivial eigenvalue yields a favourable embedding of the vertices of the hypergraph on the real axis. According to MDS-theory it is proposed in /Fuku/ to use the eigenvector according to the second-largest nontrivial eigenvalue for the location of the vertices in the second dimension of the plane. Since the eigenvectors are orthogonal, this leads to a favourable distribution of the vertices with zero-correlation. Let λ_2 and λ_3 be the largest nontrivial eigenvalues, and let $u = x^{(2)}$ and $v = x^{(3)}$ be the corresponding eigenvectors. The correlation coefficients are $\varrho_2 = \sqrt{\lambda_2}$ and $\varrho_3 = \sqrt{\lambda_3}$, respectively. In order to take into consideration the different weights of the two solutions, the eigenvectors are normalized to $\|u\| = \varrho_2$ and $\|v\| = \varrho_3$.

---

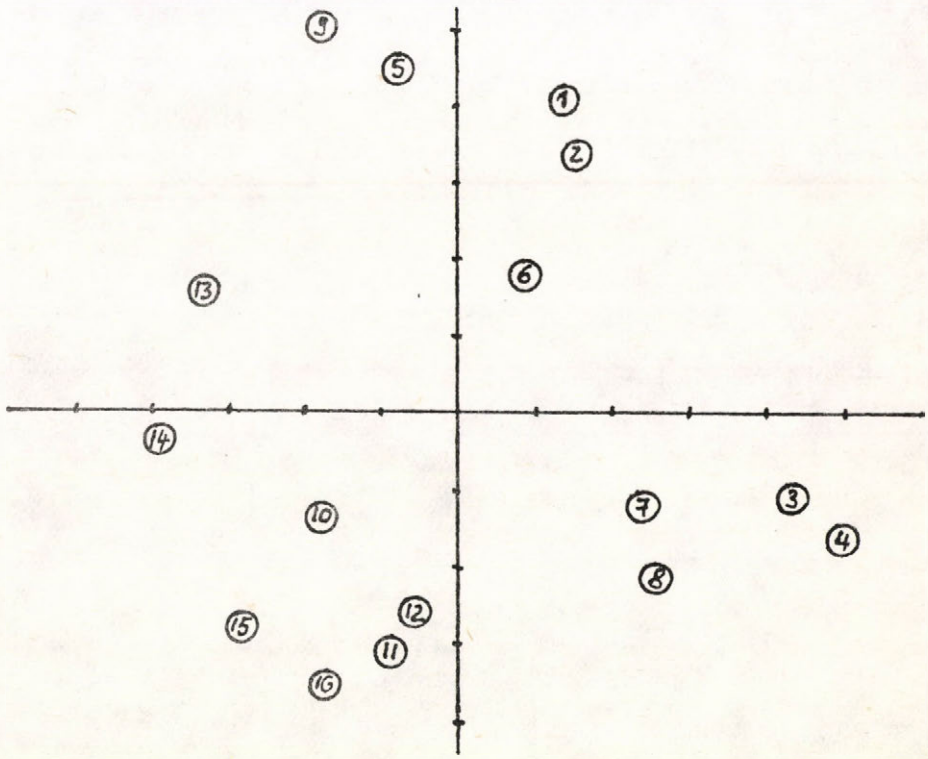
1) Notice that this problem is connected with the WEIGHTED GRAPH EMBEDDABILITY problem (cf. /Jo/) .

The set of points (u_i, v_i) - the scatter diagram - is called graphspace /Fuku/. Consider again our example. The solution of the eigenvalue problem (3.2.) yields

$\lambda_2 = 0.884$ and $\lambda_3 = 0.858$, where

$x^{(2)} = (1.29, 1.57, 4.39, 4.89, -0.85, 0.92, 2.38, 2.64,$
$-1.81, -1.79, -0.93, -0.68, -3.36, -3.90, -2.93, -1.81)$

$x^{(3)} = (+4.13, +3.32, -1.15, -1.72, +4.51, +1.76, -1.21, -2.23,$
$+5.00, -1.42, -3.15, -2.69, +1.64, -0.35, -2.77, -3.66)$



Thus we have an embedding of the hypergraph in Euclidean space, preserving in some sense the distances, i.e. reflecting the connectivity. In many applications (e.g. in gate array techniques), the set of possible module locations (slots) form a discrete and regular set of points.

An equidistant grid is a simple model of a discrete placement media which is often used. In this case the points of the scatter diagram produced by the non-discrete method have to be shifted to the grid points.

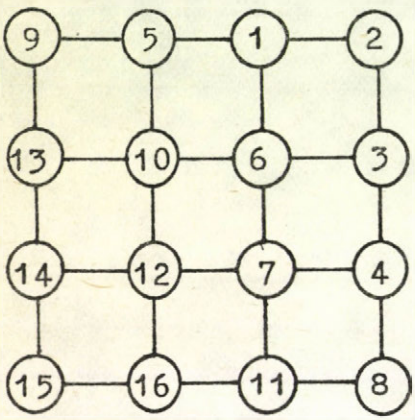
This is performed in two steps.

1. The set of points derived from the eigensolution is normalized in such a way that the first and second moments are

equal to the first and second moments of the grid, respectively.

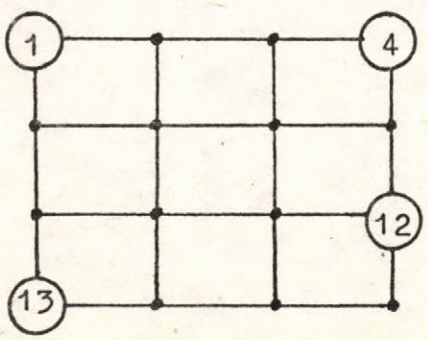
2. The points are translated to the grid points (see also /Fuku/). This step is performed with the help of a linear-assignment procedure (see e.g. /BuDe/).

For our example we obtain the following assignment to a regular (4 x 4)-grid:



5 The Placement Problem under Constraints

In most real-world placement problems we have to satisfy additional constraints as far as certain modules have a-priori fixed places on the placement media (grid). In consideration of such constraints in /Fuku/ a solution is proposed which yet disregards the connections between the fixed and the free modules. However, there is a possibility to incorporate the constraints (from the very beginning) into the eigenvalue problem. We shall explain the idea for this approach in terms of the x-direction of the grid. According to the relations between the components corresponding to the fixed modules, not all vectors $x = (x_1,\ldots,x_m)^T$ are admissable now. The admissable vectors form a subspace, which can be described as the range of a linear transformation. Without loss of generality assume that the variables $x_1,\ldots,x_q$ correspond to the a-priori fixed, and the remaining variables $x_{q+1},\ldots,x_m$ correspond to the free modules. Consider the example presented in § 4 under the following placement constraints.

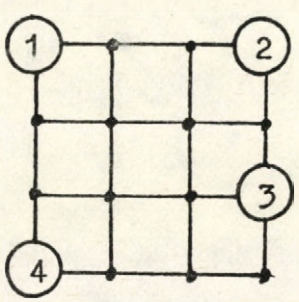


First we have to perform a re-numeration of the modules:

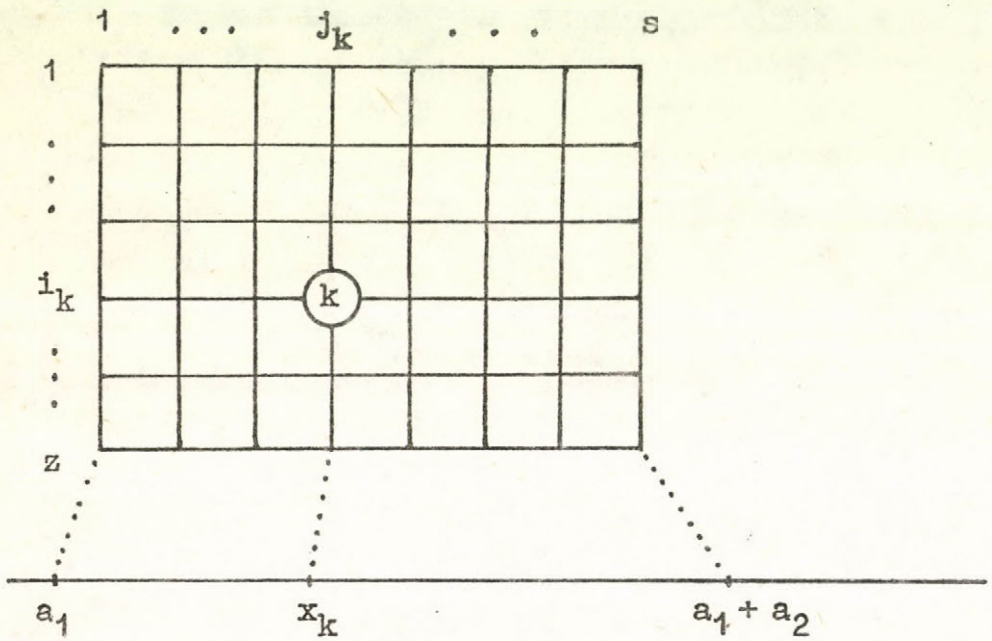
old index : 1 4 12 13 : 2 3 5 ... 11 14 15 16

new index : 1 2 3 4 : 5 6 7 ... 13 14 15 16

According to the new indices we have the following placement constraints.



With respect to the solution method using scaling theory it would not be adequate to formulate the constraints (related to the x-axis) in the form $x_1 = x_4 = 1$, $x_2 = x_3 = 4$. What we need in this model (because of the conditions $Ex = 0$ and $Var(x) = 1$) are two degrees of freedom: translation and expansion. They serve as parameters for $x_1,\ldots,x_q$. Therefore, we introduce two variables a_1 and a_2 instead of $x_1,\ldots,x_q$. Here the variable a_1 corresponds to the left border of the placement media (grid), whereas the variable a_2 corresponds to the expansion of the media.



Assume that module k is pre-assigned to the j_k-th column of the grid, and assume that the grid has s columns. Then we define

$$\sigma_k = \frac{j_k - 1}{s - 1} \qquad (k=1,\ldots,q) \,. \qquad (5.1.)$$

For our example, we have $\sigma_1 = \sigma_4 = 0$ and $\sigma_2 = \sigma_3 = 1$. Then for each x_k $(k \leq q)$ we have the condition

$$x_k = a_1 + \sigma_k \cdot a_2 \quad , \quad \text{i.e.} \qquad (5.2.)$$

$$\begin{pmatrix} x_1 \\ \vdots \\ x_q \end{pmatrix} = \begin{pmatrix} 1 & \sigma_1 \\ \vdots & \vdots \\ 1 & \sigma_q \end{pmatrix} \cdot \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \qquad (5.3.)$$

If we define $x_q^1 = \begin{pmatrix} x_1 \\ \vdots \\ x_q \end{pmatrix}$, $\Sigma = \begin{pmatrix} 1 & \sigma_1 \\ \vdots & \vdots \\ 1 & \sigma_k \end{pmatrix}$, $a = \begin{pmatrix} a_1 \\ a_2 \end{pmatrix}$,

then from (5.3.) we get

$$x_q^1 = \Sigma \cdot a \quad . \qquad (5.4.)$$

If we assume, that not all fixed modules occupy the same column, then the rank of Σ equals 2. Otherwise, this rank equals 1. In the latter case we could turn to the reduced form

$$x_q^1 = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \cdot (a_1) \quad . \qquad (5.5.)$$

In order to unify both cases, we can assume in (5.4.), that

$$a = \begin{pmatrix} a_1 \\ \vdots \\ a_t \end{pmatrix} \text{ and } \Sigma = (\sigma_{ij})_{q,t} \text{ with rank } (\Sigma) = t \quad . \qquad (5.6.)$$

Now we define

$$u = \begin{pmatrix} a_1 \\ \vdots \\ a_t \\ x_{q+1} \\ \vdots \\ x_m \end{pmatrix} \quad \text{and} \quad \Phi = \left(\begin{array}{c|c} \Sigma & 0 \\ \hline 0 & \mathfrak{E}_{m-q} \end{array}\right)$$

Instead of (5.4.) we obtain

$$x = \Phi \cdot u \quad , \quad \text{where rank } (\Phi) = m - q + t \quad . \qquad (5.7.)$$

Furthermore, we can assume (cf. (5.3.) and (5.5.)) that there exists $\tilde{e} \in \mathbb{R}_{m-q+t}$ such that

$$e_m = \Phi \cdot \tilde{e} \quad . \qquad (5.8.)$$

Now we can apply (2.6.) using the formulas (5.7.) and (5.8.) derived from the constraints. Doing this, we obtain the following optimization problem:

$$\max_{\substack{u^T(\Phi^T \cdot R_P \cdot \Phi)\cdot u=1 \\ u^T(\Phi^T \cdot R_P \cdot \Phi)\cdot \tilde{e}=0}} [u^T (\Phi^T \cdot P \cdot C_P^{-1} \cdot P^T \cdot \Phi)\cdot u] \qquad (5.9.)$$

Here the matrix R_P is a quadratic nonnegative diagonal matrix. Hence $\sqrt{R_P}$ also exists. Thus

$$\Phi^T \cdot R_P \cdot \Phi = (\sqrt{R_P}\cdot\Phi)^T \cdot (\sqrt{R_P}\cdot\Phi) \qquad (5.10.)$$

is a Gramian and obviously a positive semidefinit matrix. Moreover, since the columns of Φ are linear undependent, this is also true for $\sqrt{R_P}\cdot\Phi$. Therefore, the matrix $\Phi^T \cdot R_P \cdot \Phi$ is even a regular and in particular a positiv definit matrix.

Hence, similarly to the derivation shown in § 3, we can use the theorem about extremal points of quadratic forms. Doing this, we obtain the eigenvalue problem

$$(\Phi^T \cdot P \cdot C_P^{-1} \cdot P^T \cdot \Phi)\cdot u = (\Phi^T \cdot R_P \cdot \Phi)\cdot \lambda \cdot u \ .$$

Considering this eigenvalue problem we first remark, that - similarly to § 3 - we can write this problem in the following form:

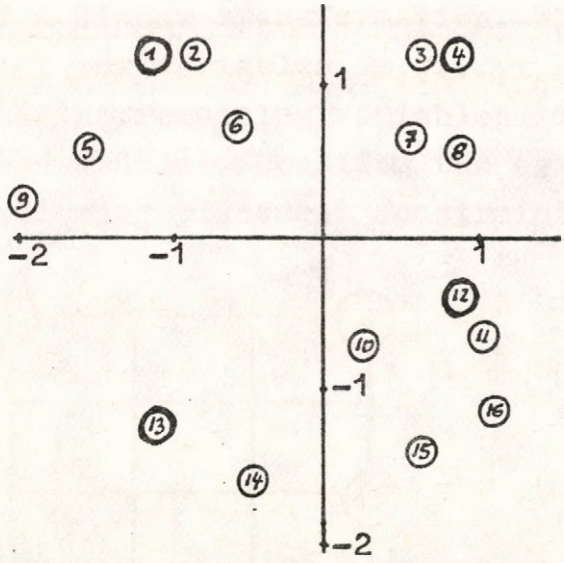
$$(\Phi^T \cdot A \cdot C^{-1} \cdot A^T \cdot \Phi)\cdot u = (\Phi^T \cdot R \cdot \Phi)\cdot \lambda \cdot u \ . \qquad (5.11.)$$

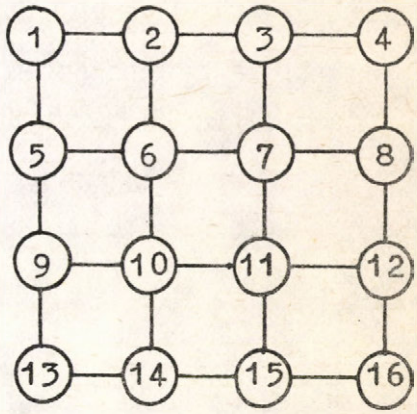
Because of the relation $e_m = \Phi \cdot \tilde{e}$, the vector $\tilde{e}$ is clearly the trivial solution corresponding to $\lambda_1 = 1$. Hence we are interested in the second-largest eigenvalue λ_2 and the corresponding eigenvector $u^{(2)}$.

From this solution, using the transformation (5.7.), we obtain the vector x containing the x-coordinates of the m modules. (Note that we have to turn back to the original enumeration of the modules.)

For our example we have $\sigma_1 = \sigma_4 = 0$ and $\sigma_2 = \sigma_3 = 1$ (cf. 5.3.). The eigenvalue problem (5.11.) then yields the solution $\lambda_2 = 0.881$ with

x = (-1.082,-0.892, 0.671, 0.882,-1.648,-0.510, 0.588, 0.936,
-1.954, 0.285, 1.004, 0.882,-1.082,-0.358, 0.676, 1.130)
(Here the order of components already corresponds to the original enumeration of modules.)
The y-coordinates of the desired grid placement can be computed in a completely analogous manner. For our example we have $\sigma_1' = \sigma_2' = 1$, $\sigma_3' = 1/3$, $\sigma_4' = 0$. The solution of the eigenvalue problem (5.11.) then yields $\lambda_2' = 0.860$ with
y = (1.247, 1.225, 1.241, 1.247, 0.500, 0.698, 0.644, 0.552,
0.205,-0.865,-0.752,-0.407,-1.235,-1.661,-1.508,-1.067).
The following figure shows the graph space together with the final solution derived from the application of a linear assignment procedure to the "free points".





We add some remarks concerning the placement procedure described in this chapter.

First notice that certain problems may arise using the procedure for special instances. As an example, assume $\sigma_i = \sigma_i'$ for $i = 1,\ldots,q$. Then both eigenvalue problems (5.11.) derived from Σ and Σ', respectively, would coinside. Hence all modules would be located on a straight line. This is also the case, if $1 - \sigma_i = \sigma_i'$ for $i=1,\ldots,q$. Moreover, if there exists an automorphism α of the hypergraph such that

$\{\alpha(1),\ldots,\alpha(q)\} = \{1,\ldots,q\}$ and

$\sigma_i = \sigma_{\alpha(i)}'$ $(i=1,\ldots,q)$ or $1 - \sigma_i = \sigma_{\alpha(i)}'$ $(i=1,\ldots,q)$,

then all free modules would be located on a straight line. Notice that we have indicated only sufficient conditions yielding pathological solutions. For practical applications we recommend the following procedure. First recognize by considering the graph space, whether the free modules possess a good distribution. This should be done by determining the correlation coefficient ρ_o of the free points in the graph space. In the case that $\rho_o \notin [-1+\varepsilon, 1-\varepsilon]$ we would recommend to take the second-largest nontrivial solution of (5.11.) (instead of the largest one) to be the locations in one of the dimensions. If then the distribution of the graph space of the free modules is not improved, then take the third-largest solution etc.

The second remark concerns the numerical solution of (5.11.) and constitutes a relation to the averaging model discussed in § 3. This remark is directed to the application of sparse-matrix techniques.
Remember that we have the problem

$$B \cdot u = \lambda \cdot u \quad , \quad \text{where}$$

$$B = (\Phi^T \cdot R \cdot \Phi)^{-1} \cdot (\Phi^T \cdot A \cdot C^{-1} \cdot A^T \cdot \Phi) \; .$$

Define

$$N_1 = (\Phi^T \cdot R \cdot \Phi)^{-1} \cdot \Phi^T \cdot A$$

$$N_2 = \qquad C^{-1} \quad \cdot (\Phi^T \cdot A)^T \; .$$

Then we can write

$$B = N_1 \cdot N_2$$

Consider the special shape of these matrices:

$$\Phi = \begin{pmatrix} \Sigma & 0 \\ 0 & \mathfrak{E} \end{pmatrix} \quad , \quad R = \begin{pmatrix} R_0 & \\ & R_1 \end{pmatrix} \begin{matrix} \} q \\ \} m-q \end{matrix}$$

$$A = \begin{pmatrix} A_0 \\ A_1 \end{pmatrix} \begin{matrix} \} q \\ \} m-q \end{matrix} \; .$$

Then we have

$$\Phi^T \cdot A = \begin{pmatrix} \Sigma^T & 0 \\ 0 & \mathfrak{k} \end{pmatrix} \cdot \begin{pmatrix} A_0 \\ \cdots \\ A_1 \end{pmatrix} = \begin{pmatrix} \Sigma^T \cdot A_0 \\ \cdots \\ A_1 \end{pmatrix} \tag{5.13.}$$

and

$$(\Phi^T \cdot R \cdot \Phi)^{-1} = \left[\begin{pmatrix} \Sigma^T & 0 \\ 0 & \mathfrak{k} \end{pmatrix} \cdot \begin{pmatrix} R_0 & 0 \\ 0 & R_1 \end{pmatrix} \cdot \begin{pmatrix} \Sigma & 0 \\ 0 & \mathfrak{k} \end{pmatrix} \right]^{-1} = \tag{5.14.}$$

$$= \begin{pmatrix} \Sigma^T \cdot R_0 \cdot \Sigma & 0 \\ \cdots & \cdots \\ 0 & R_1 \end{pmatrix}^{-1} = \begin{pmatrix} \Sigma^T \cdot R_0 \cdot \Sigma)^{-1} & 0 \\ \cdots & \cdots \\ 0 & R_1^{-1} \end{pmatrix}.$$

Hence we obtain

$$N_1 = \begin{pmatrix} (\Sigma^T \cdot R_0 \cdot \Sigma)^{-1} \cdot (\Sigma^T \cdot A_0) & 0 \\ \cdots & \cdots \\ 0 & R_1^{-1} \cdot A_1 \end{pmatrix} \tag{5.15.}$$

$$N_2 = C^{-1} \cdot ((\Sigma^T \cdot A_0)^T \vdots A_1^T) = (C^{-1} \cdot (\Sigma^T \cdot A_0)^T \vdots C^{-1} \cdot A_1^T).$$

Considering (5.15.) we see that there is an evident similarity to the formulas (3.3.). Notice that the approach presented here can also be derived using the averaging model, modified by additional conditions. Then in the alternating averaging process because of the constraints arises the problem of approximating an overdetermined systems of linear equations. Using the least squares method the eigenvalue problem (5.11.) can be derived (see /NeGoe/).

Note that the matrices N_1 and N_2 can be determined more concretely when separating between the cases

$$\Sigma = \begin{pmatrix} 1 \\ \vdots \\ 1 \end{pmatrix} \quad \text{and} \quad \Sigma = \begin{pmatrix} 1 & r_1 \\ \vdots & \vdots \\ 1 & r_q \end{pmatrix}$$

The derivation is omitted here. From the above consideration follows: In order to solve the eigenvalue problem with the help of the power method, it is necessary to store the matrices

$$\begin{pmatrix} \Sigma^T \cdot A_0 \\ \cdots\cdots \\ A_1 \end{pmatrix} \quad , \quad C, R_1 \quad \text{and} \quad (\Sigma^T \cdot R_0 \cdot \Sigma)^{-1} \; .$$

Here it may be efficient to apply sparse-matrix techniques for the matrix A_1 . Finally remark that for both the x-axis solution and the y-axis solution of the placement problem the majority of the necessary information is identical, in particular the matrices A_1 , C and R_1 . For a simultaneous solution of the eigenvalue problems this could be used in order to improve the efficiency of the whole procedure.

Acknowledgements

We are indebted to Dr. L. Telegdi and Dr. M. Simonowics (Budapest) for useful discussions contributing to this subject.

References

/Be/ Berge, C.: Graphs and hypergraphs, Amsterdam, 1973

/BuDe/ Burkard, R.E., U. Derigs: Assignment and Matching Problems: Solution Methods with FORTRAN programs, Lecture Notes in Economics and Mathem. Systems, vol. 184 (1980), Springer-Verlag, Berlin

/CMST/ Chung, M.J., F. Makedon, I.H. Sudborough, J. Turner: Polynomial time algorithms for the min cut problem on degree restricted trees, SIAM J. Comp., 14 No 1 (1985)

/Co/ Collatz, L.: Eigenwertaufgaben. Mit technischen Anwendungen, Leipzig, 1949

/Fuku1/ Fukunaga, K., S. Yamada, H. Stone, T. Kasai: Placement of Circuit Modules Using a Graph Space Approach, IEEE Proc. 20th Design Automation Conference (1983)

/Fuku2/ Fukunaga, K., S. Yamada, H. Stone, T. Kasai: A Representation of Hypergraphs in the Euclidean Space, IEEE Transactions on Computers, vol. C-33 No 4 (1984)

/Ga/ Gantmacher, F.R.: Matrizenrechnung I, Berlin, 1958

/GaJo/ Garey, M.R., D.S. Johnson: Computers and Intractability, San Francisco, 1979

/GoeNe/ Goetze, B., W. Nehrlich: Placement of Electronic Circuits under Module Constraints, Proc. 7-th European Conf. on Circuit Theory and Design, Prague 1985, 33-36

/Jo/ Johnson, D.S.: The NP-Completeness Column: An Ongoing Guide, Journal of Algorithms 3 (1982)

/KeSt/ Kendall, M., Stuart, A.: The advanced theory of statistics 2, London, 1961

/Ma Kc Bi/ Mardia, K.V., Kent, J.T., Bibby, J.M.: Multivariate Analysis, London, 1979

/May/ May, M.: Zwei Graphenalgorithmen und ihre Anwendung in der Strukturoptimierung und in der Theorie linearer Gleichungssysteme. ZKI-Inf. (1982) 4, 119-136

/Ma Me/ May,M., P. Mennecke: Layout of Schematic Drawings. Syst. Anal. Model. Simul. 1 (1984) 4, 307-338

/Ne Goe/ Nehrlich, W., B. Goetze: Äquivalenz verschiedener Ansätze zur nichtdiskreten Bearbeitung schwerer diskreter Optimierungsaufgaben, unpublished manuscript, 1985

/Otten1/ Otten, R.H.J.M.: Eigensolutions in top-down layout design, Proc. International Symposium on Circuits and Systems ISCAS (1982)

/Otten2/ Otten, R.H.J.M.: Automatic Floorplan Design, IEEE Proc. 19th Design Automation Conference (1982)

/Qui Br/ Quinn, N., M.A. Breuer: A Force Directed Component Placement Procedure for Printed Circuit Boards, IEEE Transactions on Circuits and Systems, vol. CAS-26 (1979)

/Rao/ Rao, C.R.: Linear statistical inference and its applications, New York, 1965

Scaling of random variables and arrangement problems in layout design

B. Goetze and W. Nehrlich

Summary

An optimization model from mathematical statistics - scaling of random variables by optimization of correlation coefficients - is applied to several arrangement problems for graphs and hypergraphs. The two-dimensional embedding of hypergraphs is in the centre of our attention because of its importance for the topological design of electronic circuits. For this case, the optimization model is generalized for an important case of additional constraints. The mathematical derivation of this non-discrete optimization strategy is presented in detail. Moreover, relations to similar non-discrete methods as well as some computational aspects of this approach are discussed.

Valószinüségi változók skálázása és elrendezési problémák az ültetési tervben

B. Goetze, W. Nehrlich

Összefoglaló

A szerzők a valószinüségi változók skálázására a matematikai statisztikában használt optimalizációs modellt /korrelációs együtthatók optimalizálása/ alkalmazzák gráfok és hipergráfok elrendezési problémáira. Figyelmüket leginkább a hipergráfok két-dimenziós elhelyezésére összpontositják, mert ez egy fontos feladat az elektromos áramkörök "topológiai" tervezésénél. Ebben az esetben az optimalizációs modellt ugy általánositják, hogy az korlátozó mellékfeltételek jelenléte esetében is müködjön. Az igy felmerülő nem-diszkrét optimalizálási stratégiát a szerzők részletesen is ismertetik. Ezen felül a módszert más nem-diszkrét módszerekkel is összevetik és a módszer számitástechnikai aspektusait is tárgyalják.

MTA SZTAKI Közlemények 37/1987 pp.97-116

THE CLASSES OF LANGUAGES CAN BE IDENTIFIED IN THE LIMIT FROM TEXT PRESENTATIONS

LE THANH HAI-NGUYEN QUANG MINH

Department of Computer Science
Polytechnical Institute of Hanoi, Vietnam

1. INTRODUCTION

In recent years the grammatical inference problem has been given increasing attention, not primarily in order to find specific answers to specific questions, but to study various means by which large, but related, classes of problems can be approached [1]-[9], [11], [12]. Briefly stated the problem is this: given a set of finite samples from a language, select, from a set of grammars, a grammar for the given language. Although this problem is formulated in the terminology of formal language theory, it has significant implications for the more general problem of inductive inference since it is deeply concerned with larger questions of methodology [3], [4], [6], [11], [12].

In [1] a formal model for grammatical inference was proposed and the concept of "identification in the limit" was introduced . A sufficient condition under which a class of languages can be identified in the limit, was shown in [2].

The aim of this paper is to extend the results which were presented in [2] about identification in the limit from text presentations.

Section 2 introduces a formal model for grammatical inference which is proposed by Gold [1].

In section 3 the concept of "weighted metrics" on set of languages and a special class of "weighted metrics", which are called ".effective metrics", are defined.

In section 4 the concept of "convergence" of language sequence is defined and the special convergent forms of language sequences according to weighted metrics are investigated. A necessary and sufficient condition for the convergence of a language sequence is showed and some properties of the convergent language sequences are presented among which the invariable property of the convergence of language sequences according to different weighted metrics is remarkable.

In section 5 we propose two grammar inference algorithms and show the conditions under which a class of languages can be identified in the limit from text presentations by these algorithms.

2. A MODEL FOR LANGUAGE IDENTIFICATION

Definition 2.1: An *information sequence*, I(L), of a language L, is an infinite sequence of strings from the set:

$$\{+y \mid y \in L\} \cup \{-y \mid y \in T^{+} - L\}$$

where T is the *technical* alphabet.

Definition 2.2: A *positive* information sequence, $I_p(L)$, is an information sequence of L, containing only strings of the form +y.

Negative information sequence, $I_n(L)$, is similarly defined.

Definition 2.3: An information sequence is *complete* if each string in T^{+} occurs in the sequence.

Similarly, a positive information sequence is complete if each string in L occurs in the sequence.

Definition 2.4: An *arbitrary* information sequence is one in which the strings may appear in arbitrary order.

Definition 2.5: A *sample* $S_t(I)$ of an information sequence I(L) is the unordered set:

$$S_t(I) = \{\pm y_1, \pm y_2, \ldots, \pm y_t\}$$

We distinguish a positive sample:

$$S_{p,t}(I) = \{+y_1, +y_2, \ldots, +y_t\}$$

and a negative sample:

$$S_{n,t}(I) = \{-y_1, -y_2, \ldots, -y_t\}$$

Definition 2.6: A class Γ of grammars over a finite terminal alphabet T is called *acceptable* if:

(i) Grammars in the class can be effectively enumerated
(ii) Each grammar in the class is decidable.

Definition 2.7: Let Γ be any class of grammars over a terminal alphabet T. Then a *grammar inference algorithm* is a function:

$$M_\Gamma : \{\text{finite subsets of } T^+\} \rightarrow \Gamma$$

from sets in T^+ to grammars in the class Γ.

Definition 2.8: The algorithm M_Γ *identifies* the grammar G, or the language L(G), *in the limit* if, for any arbitrary complete information sequence I(L(G)), there is a time t' such that $t > t'$ implies:

$$\text{(i)} \quad A_t = A_{t'}$$

where $A_t = M_\Gamma(S_t)$

$A_{t'} = M_\Gamma(S_{t'})$

and (ii) $L(A_{t'}) = L(G)$.

Definition 2.9: A class Γ of grammars or class $L(\Gamma)$ of languages, is called *identified in the limit* if there is an algorithm that identifies in the limit any grammar G in Γ.

Definition 2.10: A grammar A_t is called *compatible* with a sample S_t if:

$$S_{p,t} \subseteq L(A_t)$$

and

$$S_{n,t} \subseteq T^+ - L(A_t)$$

We distinguish two fundamental methods of information presentation: *text* and *information*. The *text presentation* of a language L is an arbitrary complete positive information sequence of this language and the *information presentation* is an arbitrary complete information sequence.

3. A CLASS OF METRICS ON LANGUAGES

The metrics measuring the "distances" between two languages are defined on the set of all languages Λ_U over an arbitrary finite terminal vocabulary T. That is:

$$\Lambda_U = \{L \mid L \subseteq T^+\}$$

Λ_U is called the *universal class* of languages.

Definition 3.1: Let the terminal vocabulary T be in certain fixed order. Then a unique *lexicographic order* for T^+ can be defined by the following rules:

(i) For any strings $x,y \in T^+$, if $l(x) < l(y)$ then x precedes y in T^+, where $l(x)$ is the length of the string x.

(ii) For any strings $x = a_1a_2\ldots a_n$ and $y = b_1b_2\ldots b_n$, where $x,y \in T^+$ and $l(x) = l(y) = n$, let $a_i = b_i$ for $i = 1,2,\ldots,k$ and $a_{k+1} \neq b_{k+1}$ where $0 \leq k < n$. Then x precedes y in T^+ if a_{k+1} precedes b_{k+1} in T.

Definition 3.2: Let the terminal vocabulary T be in certain fixed order. Then any language $L \in \Lambda_U$ can be uniquely expressed as a binary *membership sequence* $F_L = \langle f_1, f_2, \ldots \rangle$ where $f_i = 1$ if the i'th string in the lexicographic ordering of T^+ is in L and $f_i = 0$ otherwise.

Definition 3.3: $W = \langle w_1, w_2, \ldots \rangle$ is a sequence of *weights* if, for all $i \geq 1$, w_i is positive and rational, and $\sum_{i=1}^{\infty} w_i = 1$.

The following lemmas can be easily proved.

Lemma 3.1: Consider the universal set Λ_U on which an associative binary operation is symmetric difference $\oplus$. Let $W = \langle w_1, w_2, \ldots \rangle$ be a sequence of weights. Then the function:

$$\|L\|_w = \sum_{i=1}^{\infty} f_i w_i$$

defines a norm on Λ_U.

Lemma 3.2: Let w be a sequence of weights. Then the function

$$d_w(L_1, L_2) = \| L_1 \oplus L_2 \|_w$$

defines a metric on Λ_U.

Definition 3.4: Let T^+ have the lexicographic order $T^+ = \langle x_1, x_2, \ldots \rangle$. Then a metric d with a sequence of associated weights $W = \langle w_1, w_2, \ldots \rangle$ is called *effective* if:

(i) $w_i = w_j$ for any i,j satisfying $l(x_i) = l(x_j)$

(ii) $w_i > \sum_{k=j}^{\infty}$ for any i,j satisfying $l(x_i) < l(x_j)$

4. THE CONVERGENCE OF A SEQUENCE OF LANGUAGES CORRESPONDING TO WEIGHTED METRICS

Definition 4.1: Let d be any weighted metric on Λ_U. Then a sequence of languages in Λ_U, $\langle L_1, L_2, \ldots \rangle$, is called *convergent* to a language $L \in \Lambda_U$, *corresponding to metric d*, if: for any $\varepsilon > 0$ there exists n such that for all $k > n$:

$$d(L_k, L) < \varepsilon$$

Lemma 4.1: A sequence of languages, $\langle L_1, L_2, \ldots \rangle$, converges to L if and only if for any positive integer l there exists n such that for all $k > n$, $l(L_k \oplus L) > l$, where $l(L_k \oplus L)$ is the length of the shortest string of $L_k \oplus L$.

Proof: Let T be in certain fixed order. Then T^+ has the following lexicographic ordering:

$$T^+ = \langle x_1, x_2, \ldots, x_{n_1}, x_{n_1+1}, \ldots, x_{n_2}, \ldots, x_{n_k}, \ldots, x_{n_{k+1}}, \ldots \rangle$$

where

$$l(x_{n_k+1}) = l(x_{n_k+2}) = \ldots = l(x_{n_{k+1}}) = k+1.$$

Let l be a positive integer and: $\varepsilon = \min_{1 \leq i \leq n_{l+1}-1} w_i$.

Since $w_i > 0$ then $\varepsilon > 0$. By the convergence of the sequence $\langle L_1, L_2, \ldots \rangle$, there exists n such that for all $k > n$

$$\sum_{i=1}^{\infty} f_i^k w_i < \varepsilon = \min_{1 \leq i \leq n_{l+1}-1} w_i$$

where

$$F_{L_k \oplus L} = \langle f_1^k, f_2^k, \ldots \rangle.$$

This implies:

$$f_i^k = 0 \quad \text{for all} \quad k > n \quad \text{and}$$

$$1 \leq i \leq n_{l+1}-1$$

Conversely, for any $\varepsilon > 0$ let p be the smallest index such that: $\sum_{i=p}^{\infty} w_i < \varepsilon$. For $l = l(x_p)$, there exists n such that for all $k > n$: $l(L_k \oplus L) > l = l(x_p)$.

Then $d(L_k, L) = \| L_k \oplus L \| < \sum_{i=p}^{\infty} w_i < \varepsilon$.

Definition 4.2: A sequence of languages, $\langle L_1, L_2, \ldots \rangle$, converging to L, is called convergent from *inside* (from *outside*) if $L_i \subseteq L$ ($L_o \supseteq L$) for all $i \geq 1$.

Definition 4.3: A sequence of languages, $\langle L_1, L_2, \ldots \rangle$, converging to L, is called convergent *from upside* (*from underside*) if $\|L_i\| \geq \|L\|$ ($\|L_i\| \leq \|L\|$) for all $i \geq 1$.

It is clear that any sequence of languages, converging from inside (from outside) is also a convergent from underside (from upside) sequence. However, the converse is not true.

Lemma 4.2: If sequence of languages, $\langle L_1, L_2, \ldots \rangle$, converges to L corresponding to a weighted metric defined by an order of T, it will converge to L corresponding to metric defined by any order of T.

Proof: Let metric d have a sequence of associated weights: $W = \langle w_1, w_2, \ldots \rangle$ and T have a certain fixed order. Then T^+ has the lexicographic order $T^+ = \langle x_1, x_2, \ldots \rangle$. Let $\langle L_1, L_2, \ldots \rangle$ be a sequence of languages convering to L corresponding to d. Now let T have another order and thus, T^+ has the lexicographic order: $T^+ = \langle x'_1, x'_2, \ldots \rangle$. Let's prove that $\langle L_1, L_2, \ldots \rangle$ also converges to L corresponding to metric d' defined by this order of T.

Since $\sum_{i=1}^{\infty} w_i = 1$ then for any $\varepsilon > 0$ there exists k such that $\sum_{i=k}^{\infty} w_i < \varepsilon$. Let $l = l(x'_k)$. By the lemma 4.1, there exists n such that for all $m > n$:

$$l(L_m \oplus L) > l.$$

Then

$$d'(L_m, L) < \sum_{i=k}^{\infty} w_i < \varepsilon.$$

Lemma 4.3: If a sequence of languages, $\langle L_1, L_2, \ldots \rangle$, converges to L corresponding to metric d with the sequence of associated weights $W = \langle w_1, w_2, \ldots \rangle$, it also converges to L corresponding to any other metric d' with the sequence of associated weights $W' = \langle w'_1, w'_2, \ldots \rangle$.

Proof: Let both metric d and d' be defined by the same order of T. With this order of T, let T^+ have the lexicographic order $T^+ = \langle x_1, x_2, \ldots \rangle$.

Since $\sum_{i=1}^{\infty} w'_i = 1$, for any $\varepsilon > 0$ there exists k such that $\sum_{i=k}^{\infty} w'_i < \varepsilon$. Let $l = l(x_k)$. By the lemma 4.1, there exists n such that for all $m > n$, $l(L_m \oplus L) > l$. Then $d'(L_m, L) < \sum_{i=k}^{\infty} w'_i < \varepsilon$.

From the above two lemmas we have the following theorem:

Theorem 4.1: If a sequence of languages $\langle L_1, L_2, \ldots \rangle$ converges to L corresponidng to a weighted metric, it also converges to L corresponding to any other weighted metric.

The following theorem, similar to the theorem 4.1, reserves the convergence from upside (from underside) corresponding to the effective metric.

Theorem 4.2: If a sequence of languages $<L_1,L_2,...>$ converges from upside (from underside) to L corresponding to an effective metric it also converges from upside (from underside) to L corresponding to any other effective metric.

Proof: By the theorem 4.1, if $<L_1,L_2,...>$ converges to L corresponding to an effective metric, it also converges to L corresponding to any effective metric.

Clearly, the definition of the effective metric does not depend on the order of T. Therefore, assume that T^+ has the lexicographic order $T^+ = <x_1,x_2,...>$, and d_w, $d_{w'}$ are two distinct effective metrics corresponding to the sequence of associated weights:

$$w = <w_1,w_2,...>, \quad w' = <w'_1,w'_2,...>$$

Moreover, assume that:

$$L_i = <x_{i_1},x_{i_2},...,x_{i_k},...>$$

$$L = <x_{j_1},x_{j_2},...,x_{j_k},...>$$

and

$$||L_i||_w > ||L||_w$$

We shall show that $||L_i||_{w'} > ||L||_{w'}$

Indeed, from the definition of effective metric:

If $l(x_{i_1}) < l(x_{j_1})$ then $||L_i||_{w'} > ||L||_{w'}$

If $l(x_{i_1}) = l(x_{j_1})$, $l(x_{i_2}) = l(x_{j_2}),...,l(x_{i_k}) = l(x_{j_k})$

and $l(x_{i_{k+1}}) < l(x_{j_{k+1}})$

$(k \geq 1)$ then $w'_{i_1} = w'_{j_1}$, $w'_{i_2} = w'_{j_2}$,...,$w'_{i_k} = w'_{j_k}$ and

$$w'_{i_{k+1}} > \sum_{p=k+1}^{\infty} w'_{j_p} .$$

Therefore $\| L_i \|_{w'} > \| L' \|_{w'}$

Lemma 4.4: Let $\langle L_1, L_2, \ldots \rangle, \langle L'_1, L'_2, \ldots \rangle$ and $\langle H_1, H_2, \ldots \rangle$ be sequences of languages such that $L_n \subseteq H_n \subseteq L'_n$ and $\langle L_1, L_2, \ldots \rangle$, $\langle L'_1, L'_2, \ldots \rangle$ converge to L. Then the sequence $\langle H_1, H_2, \ldots \rangle$ also coverges to L.

Proof: For all n:

$$d(H_n, L) \leq d(H_n, L'_n) + d(L'_n, L)$$

since $L_n \subseteq H_n \subseteq L'_n$, $d(H_n, L'_n) \leq d(L_n, L'_n)$

Therefore: $d(H_n, L'_n) \leq d(L_n, L) + d(L, L'_n)$

$$d(H_n, L) \leq d(L_n, L) + 2d(L, L_n).$$

By the convergence to L of the sequences $\langle L_1, L_2, \ldots \rangle$ and $\langle L'_1, L'_2, \ldots \rangle$, $d(L_n, L) \to 0$, $d(L'_n, L) \to 0$ as $n \to \infty$.

Therefore $d(H_n, L) \to 0$ as $n \to \infty$ and the sequence $\langle H_1, H_2, \ldots \rangle$ converges to L.

5. THE IDENTIFICATION IN THE LIMIT FROM TEXT PRESENTATION

Wharton [2] has shown that a class of languages which can be identified in the limit from text presentations is a

acceptable class of languages which does not contain a convergent sequence of distinct languages or in which any convergent sequence of distinct languages converges from outside.

In this section we shall propose two grammar inference algorithms and show the conditions under which a class of languages can be identified in the limit from text presentations by these algorithms.

These conditions are weaker than Wharton's, and therefore, our results may be considered to be more general.

Algorithm 5.1: Let Γ be enumerated in the order $\Gamma = \langle G_1, G_2, \ldots \rangle$ and sample at time t is $S_t = \langle y_1, \ldots, y_t \rangle$

(i) Determine a sequence of *possible solutions* at time t:

$$\Sigma_t = \langle H_1, H_2, \ldots, H_{r(t)} \rangle$$

Σ_t is formed by the following process:

Let $\Sigma_{t-1} = \langle H'_1, H'_2, \ldots, H'_{r(t-1)} \rangle$ and the last grammar in Γ that has been examined at time $t-1$ be $G_{s(t-1)}$.

Let $\Sigma'_t = \langle H'_1, \ldots, H'_{r(t-1)}, G_{s(t-1)+1} \rangle$ then Σ_t consists of that subsequence of Σ'_t whose grammars are compatible with S_t.

If $\Sigma_t \neq \phi$, the process is complete.

If $\Sigma_t = \phi$, the next grammar in Γ, $G_{s(t-1)+2}$ is tested for compatibility with S_t. This process continues untill $\Sigma_t \neq \phi$.

At $t = 0$, $\Sigma_t = \phi$

(ii) Select a *tentative solution* A_t from the sequence of possible solutions:

Determine K_t:

Let l_1 be the length of the longest string of S_t and l_2 be the length of the longest string of K_{t-1}. Then, for $t > 1$:

$$K_t = \{x \in T^+ \mid l(x) \leq \max\{l_1, l_2\}+1\}$$

and

$$K_1 = \{x \in T^+ \mid l(x) \leq l_1\}$$

A_t is selected to be the last grammar in the sequence Σ_t which has the following property:

(1) If H_i is the grammar precedes A_t in Σ_t then $L(A_t) \cap K_t \subset L(H_i) \cap K_t$ and the inclusion is strict.

Thus, at each time step of the inference process, one or more grammars in Γ is examined and a finite subsequence Σ_t of Γ is selected as a sequence of possible solutions. This sequence consists of the grammars already examined in Γ which are compatible with S_t. Since each grammar in Γ is decidable, it is also decidable whether any grammar is compatible with the sample and so, at each time, the sequence Σ_t is effectively formed from Γ. The selection of tentative solution A_t at time t is carried out in the sequence of possible solutions at this time and it is the last grammar of the sequence, which has the property (1). Since each grammar is decidable and K_t is finite, set $L(H_i) \cap K_t$ can be effectively determined for each $H_i \in \Sigma_t$. Therefore, the selection of any grammar of such property can be absolutely carried out. Note that at each time, there is at least one grammar which has this property.

Theroem 5.1: Let Γ be an acceptable class of grammars and d be an effectively weighted metric on $L(\Gamma)$. Then the grammar inference algorithm 5.1 identifies in the limit any grammar G

in Γ from any text presentation $I(L(G))$ if $L(\Gamma)$ does not contain a sequence of distinct languages which is convergent from underside.

Proof: First, we shall prove that if $I(L)$ is a text presentation for languages L then the sequence of samples $\langle S_1, S_2, \ldots \rangle$ converges from inside to L. Indeed, for any $\varepsilon > 0$ there exists k such that $\sum_{i=k+1}^{\infty} w_i < \varepsilon$. Since the information sequence is complete, there exists t' such that for all $t \geq t'$, $L \cap \{x_1, \ldots, x_k\} \subseteq S_t \subseteq L$ with the assumption that T^+ has the lexicographic order: $\langle x_1, x_2, \ldots, x_k, \ldots \rangle$. Then for all $t \geq t'$, $d(S_t, L) \leq \sum_{i=k+1}^{\infty} w_i < \varepsilon$. That is, the sequence $\langle S_1, S_2, \ldots \rangle$ converges t. L. Moreover, the sequence converges from inside since $S_t \subseteq L$.

Now, let Γ be enumerated in the order $\Gamma = \langle G_1, G_2, \ldots \rangle$ and G_k be the first grammar in Γ such that $L(G_K) = L(G)$.

Let t_1' be the first time such that $G_k \in \Sigma_t$. Obviously, t_1' is finite and $t_1' \leq k$ since at each time step, one or more grammars in Γ are examined. Thus, for all $t \geq t_1', G_k \in \Sigma_t$.

Consider the sequence of grammars $G_1, G_2, \ldots, G_{k-1}$. Each grammar in this sequence whose language does not include L(G) will be eliminated from the sequence of possible solutions Σ_t at some certain time and never be considered again. Indeed, let G_i be one such grammar and x_p be the first string in the lexicographic order of T^+ which belongs to $L(G_k) - L(G_i)$. Let t_i be the first time at which $x_p \in S_{t_i}$. Since the information sequence is complete one, such time exists.
Obviously, at this time, G_i is not compatible with the sample and hence it will never be selected to be possible solution.

Let t_2' be the maximum of the times t_i corresponding to all grammars $G_i (i < k)$ as above. Let $t_3' = \max\{t_1', t_2'\}$.

For all $t \geq t'_3$, Σ_t has the following form:

$$\Sigma_t = \langle H_1, H_2, \ldots, H_j, G_k, \ldots, H_{r(t)} \rangle$$

where $0 \leq j < r(t)$, j does not depend on t and $L(G_k) \subset L(H_i)$. Moreover, the inclusion is strict since G_k is the first in Γ such that $L(G_k) = L(G)$.

From step (ii) of algorithm 5.1, we have $K_{t-1} \subset K_t$ and $K_{t-1} \neq K_t$. The sequence $\langle K_1, K_2, \ldots \rangle$ is a sequence of distinct languages which converges from inside to T^+. Now, let's prove that there exists a time t'_4 such that for all $t \geq t'_4$. G_k has the property (1).

If $j = 0$, G_k has the property (1).

If $j \geq 1$. For all $t \geq t'_3$, consider the grammar $H_i \in \Sigma_t$ with $1 \leq i \leq j$. We have $L(G_k) \subset L(H_i)$ and $L(G_k) \neq L(H_i)$.

Let y be the first element of T^+ which belongs to $L(H_i) - L(G_k)$ and t''_i be the first time such that $y \in K_t$. Since $\langle K_1, K_2, \ldots \rangle$ converges to T^+, one such time exists. Then for all $t \geq \max\{t_3, t''_i\}$, $L(G_k) \cap K_t \subset L(H_i) \cap K_t$ and the inclusion is strict.

Let $t'_4 = \max\{t_3, t''_1, \ldots, t''_j\}$. Then for all $t \geq t''_4$, G_k has the property (1). Continuely, let's prove that there exists the time t' such that for all $t \geq t'$, G_k is the last grammar in Σ_t which has the property (1). Indeed, assume the contrary, that such time does not exist. Then for any arbitrary great t, there is a grammar $H_t \in \Sigma_t$ such that H_t has the property (1) and H_t follows G_k in Σ_t. We must have $S_t \subseteq L(H_t) \cap K_t \subset L(G_k) \cap K_t \subseteq L(G_k) = L(G)$ and $L(H_t) \cap K_t \neq L(G_k) \cap K_t$.

By the lemma 4.4 and the convergence to L(G) of the sequence of samples, $\langle S_1, S_2, \ldots \rangle$, the sequence $\{L(H_t) \cap K_t\}$ also converges to L(G).

We have:

$$d(L(H_t),L(G)) \leq d(L(H_t),L(H_t)\cap K_t)+d(L(H_t)\cap K_t,L(G))$$

$$\leq d(K_t,T^+)+d(L(H_t)\cap K_t,L(G))$$

Since the sequence $\langle K_1,K_2,\ldots\rangle$ converges to T^+, the sequence $\{L(H_t)\}$ converges to $L(G)$. Moreover, the sequence $\{L(H_t)\}$ is a sequence of distinct languages, because any grammar, following G_k in Γ, will be eliminated from Σ_t at some finite time.

Since $L(H_t)\cap K_t \subset L(G)\cap K_t$ and from the definition of the effective metric we have $\|L(H_t)\| < \|L(G)\|$. Thus $\{L(H_t)\}$ is a sequence of distinct languages which converges from underside to $L(G)$. This contradicts to the hyposthesis.

Thus, there exists t' such that for all $t \geq t'$, G_k is the last grammar in Σ_t which has the property (1). According to the step (ii) of the algorithm 5.1:

For all $t \geq t'$ $\quad A_t = G_k \quad$ and $\quad L(G_k) = L(G)$.

We receive a stronger result in the case when it is decidable whether or not any two grammars of Γ are equivalent.

Algorithm 5.2: Let Γ be enumerated in the order $\langle G_1,G_2,\ldots\rangle$ and the sample at time t is $S_t = \langle y_1,\ldots,y_t\rangle$

(i) Determine a sequence of possible solutions at time t: $\Sigma_t = \langle H_1,\ldots,H_{r(t)}\rangle$ as step (i) of algorithm 5.1.

(ii) Select a tentative solution A_t from the sequence of possible solutions:

A_t is the last grammar in the sequence having the following property:

(2) If H_i is a grammar preceding A_t in Σ_t then

$$L(A_t) \subset L(H_i)$$

and the inclusion is strict.

Theorem 5.2: Let Γ be an acceptable class of grammars and d be an weighted metric on $L(\Gamma)$. Moreover, suppose that it is decidable whether or not any two grammars of Γ are equivalent. Then $L(\Gamma)$ can be identified in the limit from text presentations if and only if $L(\Gamma)$ does not contain a sequence of distinct languages which converges from inside.

Proof: Necessarity: See [2], theorem 2.2.3.

Sufficiency: Consider the grammar inference algorithm 5.2. We shall show that it identifies in the limit any language L in $L(\Gamma)$.

Assume that Γ enumerated in the order $\langle G_1, G_2, \ldots \rangle$ and G_k is the first grammar in Γ such that $L(G_k) = L$.

According to the proof of theorem 5.1, there exists a time t_3' such that for all $t \geq t_3'$, Σ_t has the following form: $\Sigma_t = \langle H_1, \ldots, H_j, G_k, \ldots, H_{r(t)} \rangle$, where $0 \leq j < r(t)$, j does not depend on t and $L(G_k) \subset L(H_i)$. Moreover, the inclusion is strict.

Thus, for all $t \geq t_3'$, G_k is a grammar of Σ_t which has the property (2). Now we shall show that there is a time t' such that for all $t \geq t'$, G_k is the last grammar in Σ_t which has the property (2).

Indeed, assume the contrary that such time does not exist. Then for any arbitrary great t, there is a grammar $H_t \in \Sigma_t$ such that H_t has the property (2) and H_t follow G_k in Σ_t. We have:

$$S_t \subseteq L(H_t) \subset L(G_k) = L.$$

The sequence of samples $\langle S_1, S_2, \ldots \rangle$ converges to L and by lemma 4.4 the sequence $\{L(H_t)\}$ is a sequence of distinct languages which converges from inside to L, a contradiction.

Thus, there exists t' such that for all $t \geq t'$, G_k is the last grammar in Σ_t which has the property (2).

According to the step (ii) of the algorithm 5.2 for all $t \geq t'$ $A_t = G_k$ and $L(G_k) = L$.

6. CONCLUSION

Thus, by using the class of effective weighted metrics on the set of languages and the concept of convergence from underside we have achieved results more general than Wharton's for the identification in the limit from text presentations. Simultaneously, we have developed the necessary and sufficient condition for a sequence of languages be convergent corresponding to a weighted metric. This condition express the essence of the convergence of a sequence of languages and directly leads to the invariable property of the convergence corresponding to different weighted metrics. This property, essentially show the accuracy of our approach.

REFERENCES

[1] Gold, E.M., Language identification in the limit. Information and Control 10(1967), pp. 447-474.

[2] Wharton, R.M., Grammatical inference and approximation. Technical Report, No. 51, Department of Computer Science, University of Toronto, 1973.

[3] Crespi-Reghizzi, S., The mechanical acquisition of precedence grammars, Report UCLA-ENG-7054, School of Engineering and Applied Science, University of California, 1970.

[4] Fu, K.S., Syntactic methods in pattern recognition. School of electrical engineering, Purdue University, Academic Press, New York and London, 1974.

[5] Lu, S.Y. and Fu, K.S., Stochastic Error-Correcting Syntax Analysis for Recognition of Noisy Patterns, IEEE Trans. on Computers, Vol. C-26, No. 12, 1977.

[6] Majumdar, A.K. and Roy, A.K., Inference of fuzzy regular pattern grammar, Pattern Recognition Letters, 2 (1983), pp. 27-32.

[7] Fu, K.S. and Booth, T.K., Grammatical Inference--Introduction and Survey, IEEE Trans. on Systems, Man and Cybernetics, Vol. SMC-5, Jan. and July 1975.

[8] Fu, K.S., Recent Progress in Syntactic Pattern Recognition, School of electrical engineering, Purdue University, W. Lafayette Indiana 47907, U.S.A, 1981.

[9] Thompson, R.A., Determination of probability grammars for functionally specified probability measure languages, IEEE Trans. on Computers, Vol. C-23, June 1974.

[1o] Aho, A.V., and Ullman, J.D., The theory of parsing, translation and compiling, Vol. I., Parsing, Prentice--Hall, Englewood Cliffs, New Jersey, 1972.

[11] You, K.C. and Fu, K.S., A syntactic approach to shape recognition using attributed grammars, IEEE Trans. on Systems, Man and Cybernetic, Vol. SCM-9, June 1979.

[12] You, K.C. and Fu, K.S., Distorted shape recognition using attributed grammars and error-correcting techniques, Computer Graphics and Image Processing, Vol. 13, 1980.

The classes of languages can be identified in the limit from text presentations

Le Thanh Hai, Nguyen Quang Minh

Summary

The problem of identification in the limit from text presentations is considered. A special class of weighted metrics on languages and the special convergent forms according to weighted metrics are investigated and some properties of the convergent language sequence are presented. Finally, two grammar inference algorithms are proposed and the condition of being identifiable in the limit from text presentations by these algorithms for language classes are discussed.

A nyelvek azon osztályai, amelyeket limeszben identifikálni lehet szöveg-mintákból

Le Thanh Hai, Nguyen Quang Minh

Összefoglaló

A cikkben a "limeszben" /határátmenetben/ való identifikálást ismertetik a szerzők. Bevezetnek a nyelvek között speciális, sulyozott metrikákat és azokban való konvergenciát vizsgálják. Végül két algoritmust is megadnak, amelyek segitségével feltételek adhatók meg annak eldöntésére, hogy az adott nyelv limeszben identifikálható-e szöveg-minták alapján vagy sem.

MTA SZTAKI Közlemények 37/1987 pp. 117-124

PESS - A PROLOG EXPERT SYSTEM SHELL USING INEXACT REASONING

HA HOANG HOP

Hanoi State's Committee for Science and Technology
Hanoi - Vietnam

1. INTRODUCTION

Inexact reasoning in expert systems (ESs) is one of the recent attentions in the area of designing ES. We have agreed that no single inference mechanism would appear to be more accepted in application than another. The natural consequence is to combine some inference mechanisms into an unified module which provides the user with various reasoning schemes. In this paper, PESS - a prototype of ES shell in PROLOG is presented to realize this idea. The emphasis is also aimed to PROLOG - a tentative tool for representation of uncertainty in ES. PROLOG has some significant features that aid the development of ESs. Expert system are computer programs intended to solve problems usually requiring human expertise. They typically consist of a knowledge base and an inference engine that provides mechanisms or schemes for symbolic reasoning, search and explanation.

Knowledge is usually represented in ES as production rules having the form:

IF condition THEN action or
IF antecendent THEN conclusion

One of the most common mode of inference in ES is backward changing - the input is viewed as a goal to satisfy, and the production rules are used to generate subgoals that must in turn be satisfied. The process terminates when it is asserted - either by facts in knowledge base or by the inference engine, that the goal is satisfied (succeeded) or failed.

In designing ESs, one has to consider processes of knowledge acquisition, knowledge representation, knowledge-based inference, and search-based computation along with facilities for explanation and development of expert system shells.

Building an ES shell is one of the current approaches to the problem of representing and propagating uncertainty through ESs. An ES shell can be considered as an ES without its expert knowledges, i.e., as a collection of inference mechanisms, and the facilities for providing explanation, development and debugging of the system. PESS is actually a PROLOG meta-interpreter that accounts for certainty factors by various inexact reasoning schemes.

2. INEXACT REASONING

The presence of human knowledge in an ES can be associated with at least four types of uncertainty [8]. The first type is related to the reliability of knowledge: uncertainty can be presented in factual knowledge as a result of ill-defined concepts in the observations. The second type of uncertainty is caused by the inherent imprecision of rule representation language relating to the problem of semantic matching which compares the approximate meaning of facts and premises. The third type appears when inference is based on incomplete information. And the fourth type arises from the aggregation of rules from different knowledge sources or different experts.

Here, we briefly review some inexact reasoning schemes that are used in PESS to provide a way of conbining these different types of uncertainty. We gradually consider 3 levels of representing knowledge: preliminary knowledge, rules and rules framework.

2.1. Preliminary factual knowledge

Preliminary factual knowledge is an ordered triple:

(attribute, object,value)

where value is an element of the attribute domain, that domain may be discrete or continuum set, and object may refer to as a combination of more elementary objects. A natural way to attach uncertainty to this triple is

(attribute, object, value) with CM

where CM is some certainty measure (for example, either of probability, necessity, possibility measures/distributions).

2.2. Representation of rules

Rule, in general, has the form:

IF condition THEN action or
IF antecedent THEN conclusion.

Suppose that condition (antecedent) and action (conclusion) are conjunction of triples. With this, rules are in higher levels of knowledge representation.

Uncertainity is fetched to rule by

IF condition THEN action with CM

2.3. Rule framework

In more complicated application of ES, we need more available and larger rule base, and so, we may pre-establish rule bases as framework [3],[4] with some meta-level inference.

In PRESS, we use a simple rule framework for providing the combination of certainty measures.

3. DESCRIPTION OF PESS

PESS consists of the following parts:

- Rule base in which each rule is associated with a certainty measure CM, a submodule of this rule base is served for meta-rules, in which, each meta-rule has a certainty measure that is computed by some predicates implemented in control block.

- Control block: the inference engine of PESS which uses the natural backward chaining inherited from PROLOG, besides, in this block, there exists an uncertainty handler allowing PESS to reason with different reasoning schemes (3 - valued logic reasoning [5], reasoning with threshold, EMYCIN like-reasoning [6], Bayesian). The uncertainty handler is implemented by modular and transparent manner for various reasoning schemes. See the Appendix for more details.

- User interface provides the user with window system (dialog box, pop-up menu for choosing inference mode and calls for macros/predicates to compute the combination of CM).

4. IMPLEMENTATION AND CONCLUSION

The first PESS's prototype is implemented in micro-PROLOG and for the time being, we are extending this prototype to have some debugging facilities.

This paper shows PESS - a ES shell in PROLOG that owns some good features:

(i) Different inexact reasoning schemes are combined in a single block.

(ii) The modular structure of PESS makes easy to expand with another reasoning schemes.

5. APPENDIX

The EMYCIN like-reasoning.

The rule has form:

IF antecedent THEN conclusion with CF (antecedent, conclusion), where CF is certainty factor that measures the degree to which the implication is valid.

To compute CF, we use the following expression:

CF(conclusion) = CF(antecedent) x CF(antecedent, conclusion), if CF(antecedent) > 0.2

if CF(antecedent) is less than 0.2, the rule is unappropriate, [6].

In the case, when antecedent is combined by more than one elementary antecedents Ai(i = 1,n) we use the expression below to compute CF(conclusion):

$$
\begin{aligned}
CF(\text{conclusion}) &= CF1(\text{conclusion}) + CF2(\text{conclusion}) - \\
&\quad - CF1(\text{conclusion}) \times CF2(\text{conclusion}) \\
&\quad \text{if } CF1(\text{conclusion}) >= 0 \\
&= CF1(\text{conclusion}) + CF2(\text{conclusion}) \\
&\quad + CF1(\text{conclusion}) \times CF2(\text{conclusion}) \\
&\quad \text{if } CF1(\text{conclusion}) \text{ and } CF2(\text{conclusion}) \\
&\quad \text{both less than } 0 \\
&= \frac{CF1(\text{conclusion}) + CF2(\text{conclusion})}{1 - \min\{|CF(\text{conclusion})| - |CF2(\text{conclusion})|\}} \\
&\quad \text{if } -1 < CF_1(\text{conclusion}) \times CF2(\text{conclusion}) < 0 \\
&= 1 \\
&\quad \text{if } CF1(\text{conclusion}) \times CF2(\text{conclusion}) = -1.
\end{aligned}
$$

Repeat this expression for all antecedents Ai (i=1,n) we then receive the total CF(conclusion).

In general, antecedent in this mode of reasoning can be logical combination of factual informations. So, we can compute CF through the following formulas

$$CF(F1 \text{ or } F2) = \max\{CF(F1), CF(F2)\}$$
$$CF(F1 \text{ and } F2) = \min\{CF(F1), CF(F2)\}$$
$$CF(\text{not } F) = -CF(F)$$

where F1,F2 are factual informations conjuncting to form the antecedent A.

By PROLOG, we have defined 4 predicates for and, or, not combine operations.

```
and_op(cf(A),cf(B),cf(C)) :-min(A,B,C).
or_op(cf(A),cf(B),cf(C))  :-max(A,B,C).

not_op(cf(X),cf(Y)) :- Y  is  -X.
combine_op(cf(Conclusion),cf(Antecedent),cf(Rule),
                cf(NewConclusion) :-
                Medium is Antecedent*Rule,
                c(Conclusion,Medium,NewConclusion).

c(A,B,C) :-A >= 0, B >= 0,!,C is (A+B-A*B).
c(A,B,C) :-A >= 0, B <  0,!,T is -B, min(A,T,M),
              C is (A+B)/(1-M).
c(A,B,C) :- A  <0,B>= 0,!,T  is  -A,min(T,B,M),
              C is (A+B)/(1-M).
c(A,B,C) :- A < 0,B < 0,!,C  is  (A+B+A*B).
```

Another modes of reasoning can be found in [4].

REFERENCES

[1] B.G. Buchanan, E.H. Shortliffe, Rule-based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, Addison-Wesley, Reading, MA (1984).

[2] W.F. Clocksin, C.S. Mellish, Programming in Prolog, Springer-Verlag (1981).

[3] F. Hayes-Roth et al., Building Expert Systems, Addison-Wesley, Reading, MA (1983).

[4] Ha Hoang Hop, Using PROLOG to represent uncertainty in Expert Systems, (forthcoming).

[5] S. Kleene, Introduction to Metamathematics, Van Nostrand, (1951).

[6] A.C. Scott et al., EMYCIN Manual, Stanford University (1981).

[7] G. Shafer, A Mathematical Theory of Evidence, Princeton University Press, Princeton, NJ (1976).

[8] Special Issues on Representation and Propagation of Uncertainty in Expert Systems, International Journal of Man-Machine Studies 22 (1985).

PESS - A PROLOG expert system shell using inexact reasoning

Ha Hoang Hop

Summary

This paper describes PESS - a PROLOG expert system which deals with the problem of representing and propagating uncertainty in expert systems /ESs/ in context of various inexact reasoning schemes.

PESS - egy PROLOG tipusu szakértő rendszer-váz pontatlan indoklási sémák esetében

Ha Hoang Hop

Summary

A cikkben a cimben emlitett szakértő rendszer-vázat ismerteti a szerző, különböző pontatlan indoklási sémák /reasoning schemes/ esetén. Főleg a szakértő-rendszerekben levő bizonytalanság reprezentálásának és terjedésének problémáját tárgyalja.

MTA SZTAKI Közlemények 37/1987 pp. 125-149

M-MINIMAL COVERS AND SPERNER SYSTEMS WITH APPLICATION TO THE KEY FINDING PROBLEM FOR RELATION SCHEME

PHAM THE QUE

Institute of Computer Science and Cybernetics
Hanoi - Vietnam

§1. Introduction:

In studying the relation scheme, one important problem is the determination the set of its keys. In [1] C.L. Lucchesi and S.L. Osborn gave a very interesting algorithm to find the set of all keys for any relation scheme $S=\langle \Omega, F \rangle$. Ho Thuan has some new result about keys and superkeys for a relation scheme in [4,5] and J. Demetrovics [2] proved the equivalance of candidate keys with Sperner System.

In [7] we introduced the notion of so-called M-minimal cover for a relation scheme. The necessary and sufficient condition under which a subset X of Ω, $\mathcal{M}_{\mathcal{G}}(X)$ is a M-minimal cover is established when the set of all keys was known.

Basing upon these results, in this paper we investigate the properties of M-minimal cover when a finite set Π and a Sperner System $\mathcal{S}$ on Π were given. Specially, we have established a necessary and sufficient condition for which two Sperner Systems are the set of all representative sets of each other. In other words, between the Sperner System $\mathcal{S}$ and the set of representative sets for $\mathcal{S}$ there is a close relationship and they determine each other. This means that from the given set of all keys for relation scheme we can construct the set of all its representative sets and conversely.

The set of keys for the relation scheme is just the set of all representative sets for the set of all representative sets for the set of keys.

§2. Basic definitions:

In this section we give some basic definitions. Let $H = \{a_1,\dots,a_h\}$ be a finite set. The set $\mathcal{S} = \{S_1,\dots,S_s\}$ will be called a Sperner System on the set H if it satisfies the following condition [2] :

$$\text{a)} \quad S_i \subseteq H\ ,\ S_i \nsubseteq S_j \quad \text{for } i\neq j,\ i,j = 1,2,\dots,s.$$

$$\text{b)} \quad \bigcup_{i=1}^{s} S_i = H$$

2.1: Wihout loss of generality, the ordered set H determines a matrix $\mathcal{M}_{\mathcal{S}}(H) = (\alpha_{ij})$ having h rows and s columns as follows:

$$\alpha_{ij} = \begin{cases} 1 & \text{if } a_i \in S_j \\ 0 & \text{otherwise.} \end{cases}$$

We call r_i, the i-th row of the matrix $\mathcal{M}_{\mathcal{S}}(H)$ for all $i=1,2,\dots,h$ and the following notion is used:

$$r_i \in \mathcal{M}_{\mathcal{S}}(H) \quad , \quad \mathcal{M}_{\mathcal{S}}(H) = \begin{pmatrix} r_1 \\ \vdots \\ r_h \end{pmatrix}.$$

Let us define:

$$r_i \leq r_j \iff \alpha_{ik} \leq \alpha_{jk} \qquad \begin{matrix} i,j=1,2,\dots,h \\ k=1,2,\dots,s. \end{matrix}$$

Let $X \subseteq H$ be any ordered subset of H. The subset X determines a matrix $\mathcal{M}_{\mathcal{S}}(X)$ which contains all rows r_{i_k} such that $a_{i_k} \in X$. We say that $\mathcal{M}_{\mathcal{S}}(X)$ is the submatrix of $\mathcal{M}_{\mathcal{S}}(H)$ and the meaning of the following notions are obvious:

$$\mathcal{M}_{\mathcal{G}}(X) \subseteq \mathcal{M}_{\mathcal{G}}(H), \qquad \mathcal{M}_{\mathcal{G}}(X) = \begin{pmatrix} r_{i_1} \\ \vdots \\ r_{i_k} \end{pmatrix}.$$

Let $a_j \in X$ be any element, the element a_j determines the row r_j. Let us define :

$$\mathcal{M}_{\mathcal{G}}(X) - \{r_j\} := \mathcal{M}_{\mathcal{G}}(X - \{a_j\})$$

$$\mathcal{M}_{\mathcal{G}}(H) - \mathcal{M}_{\mathcal{G}}(X) := \mathcal{M}_{\mathcal{G}}(H - X).$$

2.2: The row vector $C[\mathcal{M}_{\mathcal{G}}(X)] = (\eta_1, \ldots, \eta_s)$ is called the characteristic vector of the submatrix

$\mathcal{M}_{\mathcal{G}}(X) \subseteq \mathcal{M}_{\mathcal{G}}(H)$ if:

$$\eta_j = \begin{cases} 0 \text{ if } \sum_{i=i_1}^{i_k} \alpha_{ij} = 0 \text{ for } j=1,2,\ldots,s \\ 1 \text{ otherwise} \end{cases}$$

Where $X = \{a_{i_1}, \ldots, a_{i_k}\} \subseteq H$.

2.3: Let Q be subset of H. The submatrix $\mathcal{M}_{\mathcal{G}}(Q) \subseteq \mathcal{M}_{\mathcal{G}}(H)$ is call a M-minimal cover if it satisfies the following conditions:

a) $C[\mathcal{M}_{\mathcal{G}}(Q)] = (1,1,\ldots,1)$

b) $\bar{\exists}\, Q' \subset Q$, $\mathcal{M}_{\mathcal{G}}(Q') \subset \mathcal{M}_{\mathcal{G}}(Q)$ such that $C[\mathcal{M}_{\mathcal{G}}(Q')] = (1,1,\ldots,1)$.

If $\mathcal{M}_{\mathcal{G}}(Q)$ only satisfies the condition (a), we say that $\mathcal{M}_{\mathcal{G}}(Q)$ is a M-cover.

The set $Q \subseteq H$ is called a representative set of the System $\mathcal{G}$ if the submatrix $\mathcal{M}_{\mathcal{G}}(Q)$ determined by the subset Q is a M-minimal cover.
Let $\mathcal{Q}^{(\mathcal{G})}$ be the set of all representative sets for the System $\mathcal{G}$.

2.4: Let Y be proper subset of H $(Y \subset H)$. The set Y is called a Sp-antiset for the System $\mathcal{S}$ if it satisfies:

a) $S_i \not\subseteq Y$ for $S_i \in \mathcal{S}$

b) $\forall X: (X \subseteq H \;\&\; Y \subset X) \Rightarrow \exists S_i \in \mathcal{S}$ such that $S_i \subseteq X$.

Let $\mathcal{S}^{-1}$ be the set of all Sp-antisets for System $\mathcal{S}$. It is obvious that $\mathcal{S}^{-1}$ is also a Sperner System on H.

2.5: Example

Let $H=\{a_1,a_2,a_3,a_4,a_5,a_6\}$ and

$\mathcal{S} = \{S_1,S_2,S_3,S_4\}$.

Where $S_1=\{a_1,a_2\}$ $S_2=\{a_2,a_3,a_4\}$

$S_3=\{a_2,a_4,a_5\}$ $S_4=\{a_4,a_6\}$.

When

$$\mathcal{M}_{\mathcal{S}}(H) = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \end{pmatrix} = \begin{pmatrix} 1&0&0&0 \\ 1&1&1&0 \\ 0&1&0&0 \\ 0&1&1&1 \\ 0&0&1&0 \\ 0&0&0&1 \end{pmatrix}$$

is M-cover.

And

$$\mathcal{M}_{\mathcal{S}}(Q_1) = \begin{pmatrix} r_1 \\ r_4 \end{pmatrix} = \begin{pmatrix} 1&0&0&0 \\ 0&1&1&1 \end{pmatrix}, \quad Q_1=\{a_1,a_4\}.$$

$$\mathcal{M}_{\mathcal{S}}(Q_2) = \begin{pmatrix} r_2 \\ r_4 \end{pmatrix} = \begin{pmatrix} 1&1&1&0 \\ 0&1&1&1 \end{pmatrix}, \quad Q_2=\{a_2,a_4\}$$

$$\mathcal{M}_{\mathcal{S}}(Q_3) = \begin{pmatrix} r_2 \\ r_6 \end{pmatrix} = \begin{pmatrix} 1&1&1&0 \\ 0&0&0&1 \end{pmatrix}, \quad Q_3=\{a_2,a_6\}$$

$$\mathcal{M}_{\mathcal{S}}(Q_4) = \begin{pmatrix} r_1 \\ r_3 \\ r_5 \\ r_6 \end{pmatrix} = \begin{pmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{pmatrix}, \quad Q_4=\{a_1,a_3,a_5,a_6\}$$

are M-minimal covers and the representative sets for System $\mathcal{S}$.

$$S_1^{-1}=\{a_2,a_3,a_5,a_6\} \ , \ S_2^{-1}=\{a_1,a_4,a_3,a_5\}$$

$$S_3^{-1}=\{a_1,a_3,a_5,a_6\} \ , \ S_4^{-1}=\{a_2,a_4\}$$

are Sp-antisets for $\mathcal{S}$.

§3. The properties of the M-minimal cover

In [7] we have proved some properties of the M-minimal cover when the set of all keys for the relation scheme was given. Basing on these results, in this section, the more general properties of M-minimal covers.

First, we recall some results that have been presented in [7] .

Theorem 3.1 [7] :

Let $H=\{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S}=\{S_1,\ldots,S_s\}$ be a Sperner System on H. Then for any $a_j \in H$, there exists a set $X \subseteq H$ such that $a_j \in X$ and $\mathcal{M}_{\mathcal{S}}(X) \subseteq \mathcal{M}_{\mathcal{S}}(H)$ is a M-minimal cover.

Corollary 3.1 [7]:

Let X be any subset of H such that $\mathcal{M}_{\mathcal{S}}(X) \subseteq \mathcal{M}_{\mathcal{S}}(H)$ is a M-cover. Then there exists a set $Q \subseteq X$ such that $\mathcal{M}_{\mathcal{S}}(Q) \subseteq \mathcal{M}_{\mathcal{S}}(X)$ is a M-minimal cover.

In other words any M-cover has a M-minimal cover.

Theorem 3.2 [7] :

Let $H=\{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S}=\{S_1,\ldots,S_s\}$ be a Sperner System on H. Let any $Q \subseteq H$. Then the matrix $\mathcal{M}_{\mathcal{S}}(Q)$ is a M-minimal cover if and only if the set Q satisfies the following conditions:

a) $\forall S_i \in \mathcal{S} \Rightarrow Q \cap S_i \neq \emptyset$

b) $\forall Q' \subset Q \Rightarrow \exists S_i \in \mathcal{S}$ such that $Q' \cap S_i = \emptyset$.

The theorem 3.2 can be formulated in an another form as follows:

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = \{S_1,\ldots,S_s\}$ be a Sperner System on H. Let any $Q \subseteq H$. Then the set Q is a representative set of the System $\mathcal{S}$ if and only if the set Q satisfies both the conditions (a) and (b) .

Corollary 3.2 [7]

Let Q be any subset of H such that $\mathcal{M}_{\mathcal{S}}(Q) \subseteq \mathcal{M}_{\mathcal{S}}(H)$ is a M-minimal cover. Then for any subset $X \subseteq H$, $Q \subset X \Rightarrow \mathcal{M}_{\mathcal{S}}(X) \subseteq \mathcal{M}_{\mathcal{S}}(H)$ is a M-cover.

Corollary 3.3

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = \{S_1,\ldots,S_s\}$, $\mathcal{Q} = \{Q_1,\ldots,Q_q\}$ be a Sperner Systems on H.Then the system $\mathcal{Q}$ is the set of all representative sets for $\mathcal{S}$ if and only if the System $\mathcal{Q}$ satisfies the following conditions:

a) $\forall S_i \in \mathcal{S}$, $\forall Q_j \in \mathcal{Q} \Rightarrow S_i \cap Q_j \neq \emptyset$

b) $\forall Q_j \in \mathcal{Q}$, $\forall Q' \subset Q_j \Rightarrow S_i \in \mathcal{S}$ such that $Q' \cap S_i = \emptyset$.

Theorem 3.3 [7]

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = \{S_1,\ldots,S_s\}$ be a Sperner System on H . Let $X \subseteq H$. Then the set X is a representative set for the System $\mathcal{S}$ (i.e. $\mathcal{M}_{\mathcal{S}}(X) \subseteq \mathcal{M}_{\mathcal{S}}(H)$ is the M-minimal cover) if and only if the set H - X is a Sp-antiset for $\mathcal{S}$.

Corollary 3.4 [7]

Any Sp-antiset for $\mathcal{S}$ has the following form:

$$S^{-1} = H - \{a_{i_1},\ldots,a_{i_k}\} .$$

Where $Q = \{a_{i_1},\ldots,a_{i_k}\} \subseteq H$ is the representative set for $\mathcal{S}$.

Corollary 3.5 [7]

$$|\mathcal{S}^{-1}| = |\mathcal{Q}^{(\mathcal{S})}|$$

Where $|\mathcal{S}^{-1}|$ is a cardinality of $\mathcal{S}^{-1}$
$|\mathcal{Q}^{(\mathcal{S})}|$ is a cardinality of $\mathcal{Q}^{(\mathcal{S})}$.

Theorem 3.4

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = \{S_1,\ldots,S_s\}$ be a Sperner System on H. Let $\mathcal{Q}^{(\mathcal{S})} = \{Q_1,\ldots,Q_q\}$ be a set of all representative sets for $\mathcal{S}$. Then the set $\mathcal{Q}^{(\mathcal{S})}$ is a Sperner System on H, i.e. the set $\mathcal{Q}^{(\mathcal{S})}$ satisfies the following relations:

a) $\forall Q_i \subseteq H$, $Q_i \not\subseteq Q_j$ for all $i \neq j$, $i,j=1,\ldots,q$

b) $\bigcup_{i=1}^{q} Q_i = H$.

Proof: It is obvious that the condition (a) holds. We have to prove that $H \subseteq \bigcup_{i=1}^{q} Q_i$.

For any $a_j \in H$, by Theorem 3.1, there exists a set $Q_i \subseteq H$ such that $a_j \in Q_i$ and $Q_i \in \mathcal{Q}^{(\mathcal{S})}$, i.e. $a_j \in \bigcup_{t=1}^{q} Q_t$. Showing that $H \subseteq \bigcup_{i=1}^{q} Q_i$.
It is obvious that $H \supseteq \bigcup_{i=1}^{q} Q_i$. The condition (b) holds. The proof is complete.

Theorem 3.5

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = S_1,\ldots,S_s$ be a Sperner System on H.

Let $\mathcal{Q}^{(\mathcal{S})} = \{Q_1,\ldots,Q_q\}$ be the set of all representative sets for $\mathcal{S}$. Then $\bigcap_{i=1}^{s} S_i \neq \emptyset$ if and only if there exists a subset $\mathcal{Q}_1^{(\mathcal{S})} \subseteq \mathcal{Q}^{(\mathcal{S})}$ such that $\forall Q_j \in \mathcal{Q}_1^{(\mathcal{S})} : |Q_j| = 1$ and $\bigcap_{i=1}^{s} S_i = \bigcup_{Q_j \in \mathcal{Q}_1^{(\mathcal{S})}} Q_j$.

Proof: Suppose that $\bigcap_{i=1}^{s} S_i \neq \emptyset$. We need prove that there exists a subset $\mathcal{Q}_1^{(\mathcal{S})} \subseteq \mathcal{Q}^{(\mathcal{S})}$ such that $\forall Q_j \in \mathcal{Q}_1^{(\mathcal{S})} : |Q_j| = 1$ and $\bigcap_{i=1}^{s} S_i = \bigcup_{Q_j \in \mathcal{Q}_1^{(\mathcal{S})}} Q_j$.

Let us define $\mathcal{Q}_1^{(\mathcal{S})} = \left\{ \{a\} \mid a \in \bigcap_{i=1}^{s} S_i \right\}$. It is obvious that $|\{a\}| = 1$ and

$$\bigcap_{i=1}^{s} S_i = \bigcup_{\{a\} \in \mathcal{Q}_1^{(\mathcal{S})}} \{a\}$$

Since $a \in S_i$ for all $i=1,\ldots,s$, it follows that $\{a\}$ is the representative set of $\mathcal{S}$. This means $\{a\} \in \mathcal{Q}^{(\mathcal{S})} \Longrightarrow \mathcal{Q}_1^{(\mathcal{S})} \subseteq \mathcal{Q}^{(\mathcal{S})}$.

Conversely, let $\mathcal{Q}_1^{(\mathcal{S})} \subseteq \mathcal{Q}^{(\mathcal{S})}$ be a set that $\forall Q_j \in \mathcal{Q}_1^{(\mathcal{S})} : |Q_j| = 1$ and $\bigcup_{Q_j \in \mathcal{Q}_1^{(\mathcal{S})}} Q_j = \bigcap_{i=1}^{s} S_i$. We must prove that $\bigcap_{i=1}^{s} S_i \neq \emptyset$. By the condition (a) of the Corollary 3.3 : $\forall Q_j \in \mathcal{Q}^{(\mathcal{S})}$ and $\forall S_i \in \mathcal{S} \Longrightarrow Q_j \cap S_i \neq \emptyset$. Since $|Q_j| = 1$, it shows that $Q_j \subseteq S_i$ for every $S_i \in \mathcal{S}$ i.e.

$$Q_j \subseteq \bigcap_{i=1}^{s} S_i \neq \emptyset .$$ The proof is complete.

Corollary 3.6

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = \{S_1,\ldots,S_s\}$ be a Sperner System on H. Then $\{a_1\}, \{a_2\}, \ldots, \{a_h\}$ are the representative

sets of $\mathcal{S}$ if and only if $|\mathcal{S}| = 1$.

§4. The necessary and sufficient condition:

In this section, we will give a necessary and sufficient condition for which two given Sperner Systems on H are the set of all representative sets of each other.

Theorem 4.1:

Let $H = \{a_1,\ldots,a_h\}$ be a finite set and $\mathcal{S} = \{S_1,\ldots,S_s\}$ be a Sperner System on H. Let $\mathcal{Q}^{(\mathcal{S})} = \{Q_1,\ldots,Q_q\}$ be the set of all representative sets for $\mathcal{S}$. Then

$$\mathcal{S} \equiv \mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})}$$

In other words the set of all representative sets for the set of all representative sets for $\mathcal{S}$ is just equal to $\mathcal{S}$.

Proof: We prove that

1) $\forall S_i \in \mathcal{S} \Rightarrow S_i \in \mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})}$ i.e. the set S_i is the representative set for $\mathcal{Q}^{(\mathcal{S})}$. We need show that the set S_i satisfies the following conditions:

   a) $\forall Q_j \in \mathcal{Q}^{(\mathcal{S})} \Rightarrow S_i \cap Q_j \neq \emptyset$

   b) $\forall S' \subset S_i \Rightarrow \exists Q_j \in \mathcal{Q}^{(\mathcal{S})}$ such that $S' \cap Q_j = \emptyset$.

2) $\forall Q_i^+ \in \mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})} \Rightarrow \exists S_j \in \mathcal{S}$ such that $Q_i^+ \equiv S_j$.

   This means that the matrix $\mathcal{M}_{\mathcal{S}}(H)$ has exactly one column corresponding to the set Q_i^+.

Now let us show the statement 1.

a) Let any $S_i \in \mathcal{S}$. Since $\mathcal{Q}^{(\mathcal{S})}$ is the set of all representative sets for $\mathcal{S}$, by Corollary 3.3:

$$\forall S_i \in \mathcal{S},\ \forall Q_j \in \mathcal{Q}^{(\mathcal{S})} \Rightarrow S_i \cap Q_j \neq \emptyset.$$

b) Let any $S_i \in \mathcal{S}$ and $\forall S' \subset S_i$. We have

$(H - S') \cap S_k \neq \emptyset$ for every $S_k \in \mathcal{S}$. Assume the contrary that there exists $S_k \in \mathcal{S}$ such that $(H - S') \cap S_k = \emptyset$. It is obvious that $S_k \subseteq S' \subset S_i$ i.e. $S_k \subset S_i$. This contradicts to the definition of the Sperner System $\mathcal{S}$ on H. Since $(H - S') \cap S_k \neq \emptyset$ for every $S_k \in \mathcal{S}$, it follows that $\mathcal{M}_{\mathcal{S}}(H - S')$ is a M-cover. By Corollary 3.1, there exists $Q_j \in \mathcal{Q}^{(\mathcal{S})}$ and $Q_j \subseteq (H - S')$. Consequently, $S' \cap Q_j = \emptyset$.

Thus, we have $\mathcal{S} \subseteq \mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})}$.

Now let us prove the statement (2) .

Let any $Q_i^+ \in \mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})}$. First we show that the matrix $\mathcal{M}_{\mathcal{S}}(H - Q_i^+)$ is not a M-cover. Assume the contrary that the matrix $\mathcal{M}_{\mathcal{S}}(H - Q_i^+)$ is a M-cover. By Corollary 3.1, there exists $Q_k \subseteq (H - Q_i^+)$ such $Q_k \in \mathcal{Q}^{(\mathcal{S})}$, showing that $Q_i^+ \cap Q_k = \emptyset$. We arrive to contradiction to the fact that Q_i^+ is a representative set of $\mathcal{Q}^{(\mathcal{S})}$.

Since the matrix $\mathcal{M}_{\mathcal{S}}(H - Q_i^+)$ is not a M-cover, the characteristic vector $C[\mathcal{M}_{\mathcal{S}}[(H - Q_i^+)]$ has at least one component equal to null. Suppose its j-th component equal to null.

a) We shall prove that for all $a \in Q_i^+$, j-th component of the vector $C[\mathcal{M}_{\mathcal{S}}(\{a\})]$ must equal to 1. The proof is by contradiction. Suppose there exists $a \in Q_i^+$ such that the j-th component of the characteristic vector $C[\mathcal{M}_{\mathcal{S}}(\{a\})]$ equal to null. Consequently, the j-th component of the vector $C[\mathcal{M}_{\mathcal{S}}[(H - Q_i^+) \cup \{a\}]]$ is equal to null. On the other hand, since $Q_i^+ \in \mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})}$, it follows that the set $(H - Q_i^+)$ is a Sp-antiset of $\mathcal{Q}^{(\mathcal{S})}$. Since $(H - Q_i^+) \subset (H - Q_i^+) \cup \{a\}$, there exists $Q_k \in \mathcal{Q}^{(\mathcal{S})}$ such that $Q_k \subseteq (H - Q_i^+) \cup \{a\}$. Since Q_k is a representative set of $\mathcal{S}$ then, by Corollary 3.2, the matrix $\mathcal{M}_{\mathcal{S}}[(H - Q_i^+) \cup \{a\})$ is a M-cover.

It follows that all components of the vector $c[\mathcal{M}_{\mathcal{G}}((H - Q_i^+) \cup \{a\})]$ are equal to 1 . We arrive to a contradiction with the assumption that the j-th component of the vector $c[\mathcal{M}_{\mathcal{G}}((H - Q_i^+) \cup \{a\})]$ equal to null. Thus, we have proved that

(*) All elements in the j-th column of the matrix $\mathcal{M}_{\mathcal{G}}(Q_i^+)$ are equal to 1 . All elements in the j-th column of the matrix $\mathcal{M}_{\mathcal{G}}(H - Q_i^+)$ are equal to null.

b) Let us show that the matrix $\mathcal{M}_{\mathcal{G}}(H)$ has exactly one column which satisfies (*) . In fact, assume the contrary that there exists j-th and t-th columns which satisfy condition (*) . It is clear that $S_j = S_t$, a contradiction (by the definition of the Sperner System $\mathcal{G}$ on H) .
Combined (a) with (b) we concluded that there exists exactly one column in the matrix $\mathcal{M}_{\mathcal{G}}(H)$ that satisfies (*) and this column just the one which corresponds to set Q_i^+ , i.e. $\mathcal{Q}^{(\mathcal{G})} \subseteq \mathcal{G}$.
The proof is complete.

Corollary 4.1:

Let $H = \{a_1, \ldots, a_h\}$ be a finite set and $\mathcal{G} = \{S_1, \ldots, S_s\}$ be a Sperner System on H.Let $\mathcal{Q}^{(\mathcal{G})} = \{Q_1, \ldots, Q_q\}$ be the set of all representative sets for $\mathcal{G}$. Then the following statements are equivalent :

1) a) $\forall S_i \in \mathcal{G}, \quad \forall Q_j \in \mathcal{Q}^{(\mathcal{G})} \Longrightarrow Q_j \cap S_i \neq \emptyset$

b) $\forall Q_j \in \mathcal{Q}^{(\mathcal{G})}, \quad \forall Q' \subset Q_j \Longrightarrow \exists S_i \in \mathcal{G}$ such that $Q' \cap S_i = \emptyset$.

2) a) $\forall S_i \in \mathcal{G}, \quad \forall Q_j \in \mathcal{Q}^{(\mathcal{G})} \Longrightarrow Q_j \cap S_i \neq \emptyset$

b) $\forall S_i \in \mathcal{G}, \quad \forall S' \subset S_i \Longrightarrow \exists Q_j \in \mathcal{Q}^{(\mathcal{G})}$ such that $S' \cap Q_j = \emptyset$.

Corollary 4.2:

Let $H = \{a_1, \ldots, a_h\}$ be a finite set and $\mathcal{S} = \{S_1, \ldots, S_s\}$, $\mathcal{Q}^{(\mathcal{S})} = \{Q_1, \ldots, Q_q\}$ be a Sperner system on H. Then the necessary and sufficient condition for $\mathcal{S}$ and $\mathcal{Q}^{(\mathcal{S})}$ systems are the set of all representative sets of each other is that the systems $\mathcal{S}$ and $\mathcal{Q}^{(\mathcal{S})}$ satisfy either statement (1) or statement (2).

Example 4.1:

Let $H = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$
and $\mathcal{S} = \{S_1, S_2, S_3, S_4, S_5\}$

Where $S_1 = \{a_3, a_6, a_7\}$, $S_2 = \{a_1, a_4, a_5, a_7\}$,

$S_3 = \{a_1, a_3, a_5, a_7\}$, $S_4 = \{a_2, a_5, a_7\}$,

$S_5 = \{a_2, a_6, a_7\}$.

Then

$$\mathcal{M}_{\mathcal{S}}(H) = \begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

$$\mathcal{Q}^{(\mathcal{S})} = \{Q_1, Q_2, Q_3, Q_4, Q_5, Q_6\},$$

where $Q_1 = \{a_7\}$, $Q_2 = \{a_2, a_3, a_5\}$,

$Q_3 = \{a_5, a_6\}$, $Q_4 = \{a_1, a_2, a_3\}$,

$Q_5 = \{a_1, a_2, a_6\}$, $Q_6 = \{a_2, a_3, a_4\}$

is the set of all representative sets for $\mathcal{S}$.

And

$$\mathcal{M}_{\mathcal{Q}^{(\mathcal{S})}}^{(H)} = \begin{pmatrix} 0&0&0&1&1&0 \\ 0&1&0&1&1&1 \\ 0&1&0&1&0&1 \\ 0&0&0&0&0&1 \\ 0&1&1&0&0&0 \\ 0&0&1&0&1&0 \\ 1&0&0&0&0&0 \end{pmatrix}$$

$$\mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})} = \{Q_1^*, Q_2^*, Q_3^*, Q_4^*, Q_5^*\},$$

where

$$Q_1^* = \{a_3, a_6, a_7\} \equiv S_1 \ , \ Q_2^* = \{a_1, a_4, a_5, a_7\} \equiv S_2 \ ,$$
$$Q_3^* = \{a_1, a_3, a_5, a_7\} \equiv S_3 \ , \ Q_4^* \equiv \{a_2, a_5, a_7\} \equiv S_4 ,$$
$$Q_5^* = \{a_2, a_6, a_7\} \equiv S_5 \ .$$

It is obvious that $\mathcal{Q}^{(\mathcal{Q}^{(\mathcal{S})})}$ is the set of all representative set for $\mathcal{Q}^{(\mathcal{S})}$.

§5. Algorithms:

In this section, we present the algorithm to find the set of all representative sets of any Sperner system $\mathcal{S}$ on H, and the algorithm to recognize whether a given set $X \subseteq H$ is or is not a representative set of $\mathcal{S}$.

Remark: The algorithm to determine whether $\mathcal{M}_{\mathcal{S}}(X)$ is a M-cover is a simple matrix algorithm, so here we omit its.

5.1 Algorithm 1 :

This is an algorithm for the recognition whether a given set $X \subseteq H$ is a representative set for $\mathcal{S}$.

The block schema of the Algorithm 1 is presented in Fig. 5.1

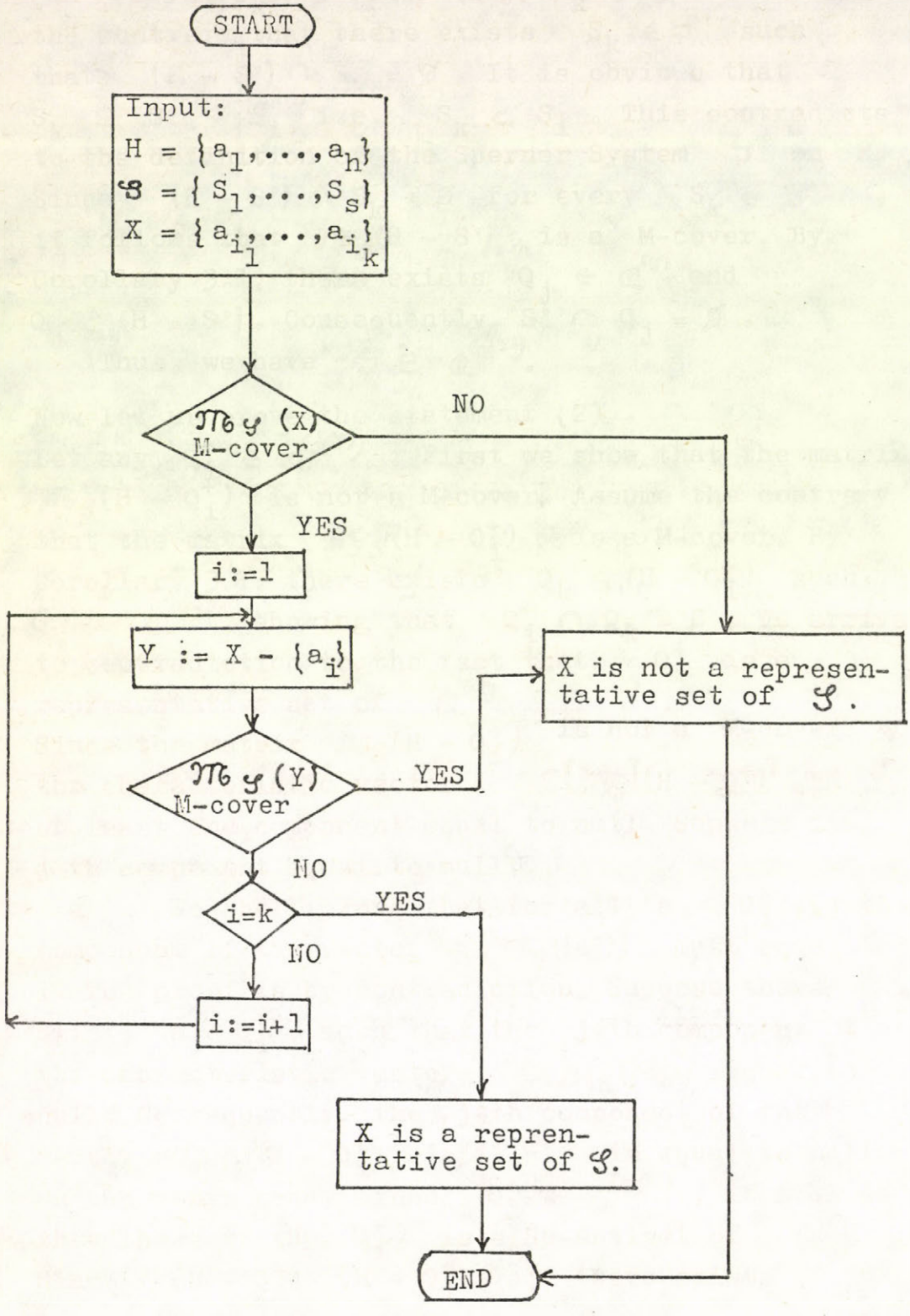

START
Input:
H = {a1,...,ah}
S = {S1,...,Ss}
X = {ai1,...,aik}
M S (X) M-cover
NO
YES
i:=1
Y := X - {ai}
X is not a representative set of S.
M S (Y) M-cover
YES
NO
i=k
YES
NO
i:=i+1
X is a representative set of S.
END

Fig. 5.1

5.2: Algorithm 2:

This is an algorithm to determine the set of all representative sets for $\mathcal{S}$ System.

Input: $H = \{a_1,\ldots,a_h\}$ is the finite set

$\mathcal{S} = \{S_1,\ldots,S_s\}$ is the Sperner System on H.

Output: The set $Q^{(\mathcal{S})}$ is the set of all representative sets for $\mathcal{S}$.

Method:

I) Let $i := 1$, $H^{(1)} := H = \{a_1,\ldots,a_h\}$,

$$\mathcal{S}^{(1)} = \mathcal{S} = \{S_1,\ldots,S_s\},$$

$$\mathcal{M}^{(1)} := \mathcal{M}_{\mathcal{S}}(H) = \begin{pmatrix} r_1 \\ \vdots \\ r_h \end{pmatrix} .$$

Let $Q := \emptyset$ will be a representative set.

II) Suppose that at i-th step, we have

$$H^{(i)} = \{a_{l_1},\ldots,a_{l_t}\}$$
$$\mathcal{S}^{(i)} = \{S^{(i)}_{j_1},\ldots,S^{(i)}_{j_k}\}$$

and

$$\mathcal{M}^{(i)} = \begin{pmatrix} r^{(i)}_{l_1} \\ \vdots \\ r^{(i)}_{l_t} \end{pmatrix}$$

$$|r^{(i)}_{l_1}| \geq |r^{(i)}_{l_2}| \geq \cdots \geq |r^{(i)}_{l_t}| > 0 .$$

The set Q is representative set of $\mathcal{S} \setminus \mathcal{S}^{(i)}$.

Case 1: If $\mathcal{M}^{(i)}$ is a M-cover.

a) If $|r^{(i)}_{l_1}| = |\mathcal{S}^{(i)}| = k$ then $\{a_{l_1}\},\ldots,\{a_{l_n}\}$ are the representative sets of $\mathcal{S}^{(i)}$ and

$$Q := Q \cup \{a_{l_1}\}, \quad Q := Q \cup \{a_{l_2}\}, \ldots, \quad Q := Q \cup \{a_{l_n}\}$$

are the representative sets for $\mathcal{S}$, where

$|r^{(i)}_{l_1}| = \ldots = |r^{(i)}_{l_n}| = k$. Hence we only consider

$$H^{(i)} := H^{(i)} - \bigcup_{j=l_1}^{l_n} \{a_j\} \quad ,$$

$$\mathcal{S}^{(i)} := \mathcal{S}^{(i)} \quad ,$$

$$\mathcal{M}^{(i)} := \mathcal{M}_{\mathcal{S}^{(i)}}(H^{(i)}) \, ,$$

$$|r^{(i)}_{l_{n+1}}| \geq \ldots \geq |r^{(i)}_{l_t}| > 0 \, .$$

b) If there exists $\mathcal{S}'' \subseteq \mathcal{S}^{(i)}$ such that $|S^{(i)}_j| = 1$ for all $S^{(i)}_j \in \mathcal{S}''$ then $\bigcup_{S^{(i)}_j \in \mathcal{S}''} S^{(i)}_j = \bigcap_{Q_q \in \mathcal{Q}^{(i)}} Q_q$.

I.e. $\bigcup_{S^{(i)}_j \in \mathcal{S}''} S^{(i)}_j$ is a common part of all the representative sets for $\mathcal{S}^{(i)}$.

Hence, we only consider :

$$H^{(i+1)} := H^{(i)} - \bigcup \{a\}$$

$$a \in S^{(i)}_j \, , |S^{(i)}_j| = 1 \, , S^{(i)}_j \in \mathcal{S}'' \, ,$$

$$\mathcal{S}^{(i+1)} := \mathcal{S}^{(i)} - \mathcal{S}'' \, ,$$

and $$\mathcal{M}^{(i+1)} := \mathcal{M}_{\mathcal{S}^{(i+1)}}\left(H^{(i+1)}\right) = \begin{pmatrix} r^{(i+1)}_{j_1} \\ \vdots \\ r^{(i+1)}_{j_g} \end{pmatrix} \, ,$$

$$|r^{(i+1)}_{j_1}| \geq \ldots \geq |r^{(i+1)}_{j_g}| \geq 0 \, .$$

$Q := Q \cup A$ is the representative set of

$\mathcal{S} - \mathcal{S}^{(i+1)}$, where $A := \bigcup \{a\}$

$$a \in S^{(i)}_j \, , \, S^{(i)}_j \in \mathcal{S}''$$

c) If either $|r^{(i)}_{l_1}| < |\mathcal{S}^{(i)}|$ or $\exists \; \mathcal{S}'' \subseteq \mathcal{S}^{(i)}$ such that $|S^{(i)}_j| = 1$, $S^{(i)}_j \in \mathcal{S}''$ then we

construct the matrix $\mathcal{M}^{(i+1)}$ as follows:

$$H^{(i+1)} := H^{(i)} - \{ a_j \mid a_j \in H^{(i)},\ r^{(i)}_j \leq r^{(i)}_{l_1},$$

where a_j determine the row $r^{(i)}_j$,

for all $j = l_2, \ldots, l_t \}$.

$$\mathcal{S}^{(i+1)} := \mathcal{S}^{(i)} - \mathcal{S}'', \text{ where}$$

$$\mathcal{S}'' := \{ S^{(i)}_j \mid S^{(i)}_j \in \mathcal{S}^{(i)},\ a_{l_1} \in S^{(i)}_j \}.$$

$$\mathcal{M}^{(i+1)} := \begin{pmatrix} r^{(i+1)}_{p_1} \\ \vdots \\ r^{(i+1)}_{p_m} \end{pmatrix}$$

$$\left| r^{(i+1)}_{p_1} \right| \geq \ \ldots \ \geq \left| r^{(i+1)}_{p_m} \right| \geq 0 .$$

And $Q := Q \cup \{ a_{l_1} \}$ is the representative set of $\mathcal{S} \setminus \mathcal{S}^{(i+1)}$.

d) Let us go to the $i := i+1$ -th step.

Case 2: If $\mathcal{M}^{(i)}$ is not a M-cover, i.e. either

$$C[\mathcal{M}^{(i)}] = (0, \ldots, 0) \quad \text{or} \quad C[\mathcal{M}^{(i)}] \neq (1, \ldots, 1).$$

a) If $i=1$, the algorithm stop.

b) If $i > 1$, let $i := i - 1$, and

$Q := Q - \{ a_{l_1} \}$. We consider a matrix

$$\mathcal{M}^{(i)} := \mathcal{M}^{(i)} - \{ r^{(i)}_{l_1} \} := \mathcal{M}_{\mathcal{S}^{(i)}} \left(H^{(i)} - \{ a_{l_1} \} \right)$$

$$\left| r^{(i)}_{l_2} \right| \geq \left| r^{(i)}_{l_3} \right| \geq \ \ldots \geq \left| r^{(i)}_{l_t} \right| > 0 .$$

Go to the case 1.

Theorem 5.1:

Let $H = \{a_1,\dots,a_h\}$ be finite set and $\mathcal{S} = \{S_1,\dots,S_s\}$ be a Sperner System on H. Then the algorithm 2 make a clearn sweep all the representative sets for $\mathcal{S}$.

Proof: Let $\mathcal{Q}^*$ be a set of all sets determined by the algorithm 2. We must prove that

1) $\forall Q \in \mathcal{Q}^* \Longrightarrow Q \in \mathcal{Q}^{(\mathcal{S})}$

2) $\forall Q \in \mathcal{Q}^{(\mathcal{S})} \Longrightarrow Q \in \mathcal{Q}^*$.

Now, we prove (1):

Let any $Q \in \mathcal{Q}^*$, $Q := \{a_{i_1},\dots,a_{i_k}\}$.

a) It is obvious that $\forall S_j \in \mathcal{S} : S_j \cap Q \neq \emptyset$

b) $\forall a_{i_j} \in Q$, $\exists S_n \in \mathcal{S}'' := \{S_t \mid a_{i_j} \in S_t\}$ such that $(Q - \{a_{i_j}\}) \cap S_n = \emptyset$.

2) We need prove that $Q \in \mathcal{Q}^{(\mathcal{S})} \Rightarrow Q \in \mathcal{Q}^*$

a) If the set Q satisfies either $|\mathcal{M}_{\mathcal{S}}(Q)| = \infty$ or $|r_{i_1}| = \dots = |r_{i_k}| = 1$, it is obvious that $Q \in \mathcal{Q}^*$.

b) Since $Q \in \mathcal{Q}^{(\mathcal{S})}$, there exists $X \subseteq H$ such that $Q \subseteq X$ and a maximal row[1] of a matrix $\mathcal{M}_{\mathcal{S}}(Q)$ is just a maximal row of a matrix $\mathcal{M}_{\mathcal{S}}(X)$. Suppose, it is r_{t_1} .

Let $X^{(1)} = X$, $Q^{(1)} = Q$, $\mathcal{S}^{(1)} = \mathcal{S}$,

$\mathcal{S}'' = \{S_j \mid S_j \in \mathcal{S}^{(1)}$, $a_{t_1} \in S_j$, a_{t_1} determine the row $r_{t_1}\}$.

Let $\mathcal{S}^{(2)} = \mathcal{S}^{(1)} \setminus \mathcal{S}''$.

Since Q is the representative set of $\mathcal{S}$, there exists $X^{(2)} \subset X^{(1)}$ such that

$Q^{(2)} := \left(Q^{(1)} - \{a_{t_1}\}\right) \leq X^{(2)}$ and the maximal row of a matrix $\mathcal{M}_{g(2)}(Q^{(2)})$ is just the maximal row of a matrix $\mathcal{M}_{g(2)}(X^{(2)})$.

Since the set Q has k elements, it follows that there exists $p > 0$ such that:

$$Q^{(1)} \supset Q^{(2)} \supset \ldots \supset Q^{(p-1)} \supset Q^{(p)} = \emptyset$$

$$X^{(1)} \supset X^{(2)} \supset \ldots \supset X^{(p-1)} \supset X^{(p)} \geq \emptyset$$

and $Q^{(m)} \subseteq X^{(m)}$ $m=1,\ldots p$.

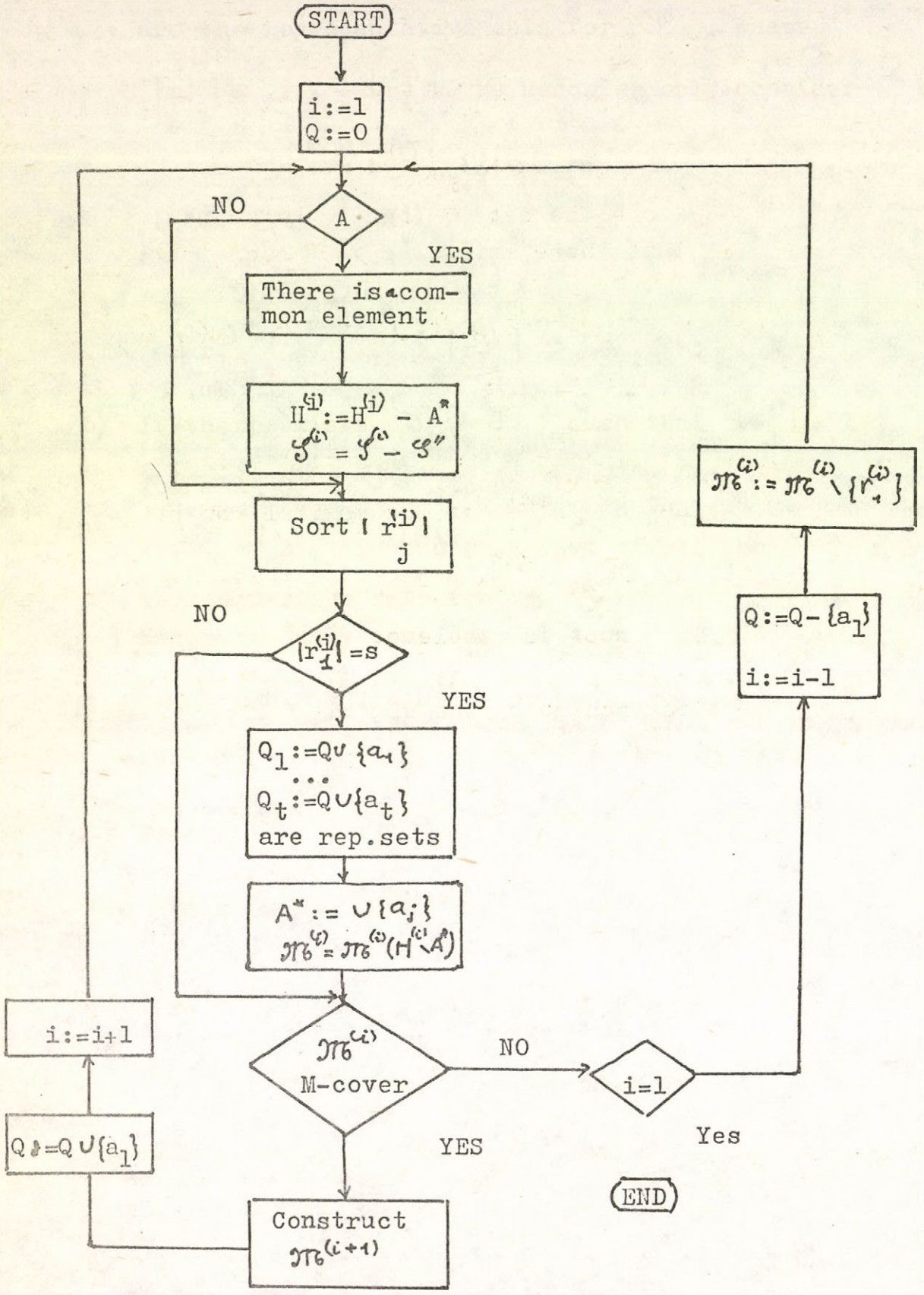
The row $r^{(m)}_{t_j}$ is the maximal row of $\mathcal{M}_{g(m)}(Q^{(m)})$ and $\mathcal{M}_{g(m)}(X^{(m)})$. It follows that $Q \in Q^*$

The proof is complete.

The block schema of the algorithm 2 is presented in Fig. 5.2 .

---

1) The row r_i is called the maximal row of the matrix $\mathcal{M}_g(X)$ if $|r_i| = \max\{|r_1|, \ldots, |r_t|\}$.



$$AC := \{ \exists\, \mathcal{G}'' \subset \mathcal{G}^{(i)} : \forall S_j \in \mathcal{G}'' \; |S_j| = 1 \}$$

Fig. 5 2

We close our paper with an example.

5.3 Example:

Let $H = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$ and

$\mathcal{S} = \{S_1, S_2, S_3, S_4, S_5, S_6\}$

where

$S_1 = \{a_7\}$, $S_2 = \{a_2, a_3, a_5\}$, $S_3 = \{a_5, a_6\}$,

$S_4 = \{a_1, a_2, a_6\}$, $S_5 = \{a_1, a_2, a_3\}$, $S_6 = \{a_2, a_3, a_4\}$.

$$\mathcal{M}_{\mathcal{S}}(H) = \begin{array}{c} \\ r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \end{array} \begin{array}{c} \begin{array}{cccccc} S_1 & S_2 & S_3 & S_4 & S_5 & S_6 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}.$$

a_7 is a common element of all the representative sets for $\mathcal{S}$. Hence , we consider only $H^{(1)} = H \setminus \{a_7\}$ and $\mathcal{S}^{(1)} = \mathcal{S} - \{S_1\}$.

$$|r_2^{(1)}| \geq |r_3^{(1)}| \geq |r_1^{(1)}| \geq |r_5^{(1)}| \geq |r_6^{(1)}| \geq |r_4^{(1)}| > 0 .$$

$r_2^{(1)}$:

$$\mathcal{M}_{\mathcal{S}^{(2)}}(H^{(2)}) = \begin{array}{c} \\ r_5^{(2)} \\ r_6^{(2)} \end{array} \begin{array}{c} \begin{array}{c} S_3 \end{array} \\ \begin{pmatrix} 1 \\ 1 \end{pmatrix} \end{array}$$

$Q_1 := \{a_7, a_2, a_5\}$, $Q_2 := \{a_7, a_2, a_6\}$.

We consider

$$\mathcal{M}_{\mathcal{S}^{(1)}}(H^{(1)} \setminus \{a_2\}) = \begin{array}{c} \\ r_1^{(2)} \\ r_3^{(2)} \\ r_4^{(2)} \\ r_5^{(2)} \\ r_6^{(2)} \end{array} \begin{array}{c} \begin{array}{ccccc} S_2 & S_3 & S_4 & S_5 & S_6 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$r_3^{(1)}$:

$$\mathfrak{M}_{\mathcal{G}^{(2)}}(H^{(2)}) = \begin{array}{c} \\ r_1^{(2)} \\ r_5^{(2)} \\ r_6^{(2)} \end{array} \begin{array}{c} \begin{array}{cc} S_3 & S_4 \end{array} \\ \begin{pmatrix} 0 & 1 \\ 1 & 0 \\ 1 & 1 \end{pmatrix} \end{array}$$

$Q_3 := \{a_7, a_3, a_1, a_5\}$, $Q_4 := \{a_7, a_3, a_6\}$.

$$\mathfrak{M}_{\mathcal{G}^{(1)}}(H^{(1)} - \{a_2, a_3\}) := \begin{array}{c} \\ r_1^{(1)} \\ r_4^{(1)} \\ r_5^{(1)} \\ r_6^{(1)} \end{array} \begin{array}{c} \begin{array}{ccccc} S_2 & S_3 & S_4 & S_5 & S_6 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \end{pmatrix} \end{array}$$

$r_1^{(1)}$:

$$\mathfrak{M}_{\mathcal{G}^{(2)}}(H^{(2)}) = \begin{array}{c} \\ r_4^{(2)} \\ r_5^{(2)} \\ r_6^{(2)} \end{array} \begin{array}{c} \begin{array}{ccc} S_2 & S_3 & S_6 \end{array} \\ \begin{pmatrix} 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 0 \end{pmatrix} \end{array}$$

$Q_5 := \{a_7, a_1, a_4, a_5\}$.

$\mathfrak{M}_{\mathcal{G}^{(1)}}(H^{(1)} - \{a_1, a_2, a_3\})$ is not a M-cover .

Thus, we have the set of all representative sets for $\mathcal{G}$:

$Q_1 = \{a_7, a_2, a_5\}$, $Q_2 = \{a_7, a_2, a_6\}$, $Q_3 = \{a_1, a_7, a_3, a_5\}$,

$Q_4 = \{a_7, a_3, a_6\}$, $Q_5 = \{a_7, a_1, a_4, a_5\}$, .

Acknowledgements:

The author would like to thank Prof.Dr. Janos Demetrovics and Dr. Ho Thuan for their valuable remarks and suggestions.

References:

[1] Lucchesi,C.L., Osborn,S.L.;Candidate keys for relation. J. of Computer and System Sciences, 17 /1978/, 270-279.

[2] Demetrovics, J.; On the equivalence of candidate keys with Sperner systems. Acta Cybernetica Szeged, 4 /1979/, 247-252.

[3] Demetrovics, J.,Ho Thuan; Some additional properties of keys for relation scheme, MTA SZTAKI, Közlemények 34/1986,37-50.

[4] Ho Thuan ; Contribution to the theory of relational databases. MTA SZTAKI, Tanulmanyok 184/1986.

[5] Ho Thuan; Some remarks on the algorithm of Lucchesi and Osborn, MTA SZTAKI, Közlemenyek 35/1986. / to appear /

[6] Ho Thuan and Le Van Bao; Some results about keys for relational schemas, Acta Cybernetica Szeged, 7 / 1985 /, 99-113.

[7] Pham The Que; The relation between antikeys and M-minimal covers in the relation schemes, MTA SZTAKI, közlemenyek 36 /1986. /to appear/

M-minimal covers and Spredner systems with application to the key finding problem for relation scheme

Phan The Que

Summary

Based on results in [7], in this paper the properties of M-minimal covers when a finite set H and Sperner system $\mathcal{S}$ on H are given are investigated. Specially, the necessary and sufficient condition for which two Sperner systems are sets of all representative sets of each other are established.

This means that from the given set of all keys for a relation scheme , its set of all representative sets can be constructed and conversely, from the set all representative sets for the set of keys, the set of keys for the relation scheme can be determined.

The set of keys the relation scheme is just the set of all representative sets for the set of keys.

M-minimális lefedések, Sperner-rendszerek és alkalmazásuk a relációs sémák kulcs-keresési problémájára

Phan The Que

Összefoglaló

A [7] eredményeire alapozva, a szerző az M-minimális lefedések tulajdonságait vizsgálja adott véges H halmaz és rajta egy Sperner-rendszer esetén. Annak szükséges és elégséges feltételét is megadja, hogy két Sperner-rendszer egymásnak teljes reprezentáló rendszerét alkotják. Ennek segitségével, ha adva van egy reláció séma kulcsainak halmaza, meg lehet konstruálni a séma teljes reprezentáló halmazát és forditva, ha adva van a teljes reprezentáló halmaz, akkor a kulcsok halmazát lehet meghatározni.
A reláció séma kulcsainak halmaza tehát semmi más, mint a kulcsok halmazát reprezentáló teljes halmaz.

MTA SZTAKI Közlemények 37/1987 pp- 151-170

A RESTRICTED DESIGN METHODOLOGY TO ALLOW TESTING FOR BCNF IN POLYNOMIAL TIME

F. N. SPRINGSTEEL

University of Missouri at Columbia
Department of Computer Science
Columbia, Missouri, 65211 USA

ABSTRACT

Boyce-Codd Normal Form (BCNF) is a well-known condition on relational database schemas that implies some desirable properties, and is known to prevent some very undesirable "anomalies" from occurring in the use of the database. It is thus important to be able to test a proposed database design for this condition.

This investigation of logical database design methods asks whether there are any conditions on a relational database schema such that, although they are not equivalent to BCNF, do guarantee that this desirable property can be detected in polynomial time (P-time). Conditions equivalent to BCNF are known to be intractable to test, but the conditions we give here are either sufficient for BCNF or else enable its being tested in P-time, and the conditions themselves are likewise testable. They are also desirable for certain design reasons.

While the first set of conditions only logically imply BCNF, they provide a setting (regular entity-relationship diagrams) which helped to suggest the second set of conditions, in which it is easier to test for this normal form than in general databases. Also, we argue, this setting is valuable in its own right, and can lead to a useful, if not universal, database design methodology. This methodology works for many databases expressible by E-R diagrams which have a "database key".

I. INTRODUCTION AND PRELIMINARIES

A. THE PROBLEM AND THE APPROACHES

It is clearly futile to search for conditions that are equivalent to BCNF, because it is an NP-complete problem to test relations for BCNF and conditions very similar to it, in general relational databases [BB79; JF; Osb]. Only very strong conditions seem to imply BCNF, e.g. 4NF,PJNF [LePa], and Berge-acyclicity [JNS83a].

However, it is possible that certain conditions related to "BCNF-ness" are so akin to this normal form that it can be tested in their presence, in tractable time. Indeed we find that this is the case; there are arguably quite reasonable, realistic conditions that one might like to see in the schema design normally, fitting the above description.

For now we shall only describe one set of conditions in intuitive terms : later on we will give their formal descriptions. These conditions model a type of very well designed database: there is one master-file, S, which contains either a key or a determiner of a key for each of the other files. These other files relate either one key to its directly dependent attributes or else relate several equivalent keys, possibly of different entities. We refer to databases of this type as fitting our "master-file" scenario.

In the next part of this first section we review the special concepts needed for our approach, including the definition of "regular" for an entity-relationship diagram (ERD). We assume that the reader is familiar with standard relational database theory, including the concepts of functional dependencies, keys, and schema [Ulm82a].

In Section II we discuss conditions that are sufficient to imply BCNF for relation schemes in the canonical relational schema |R of a regular ERD |E. Certain of these schemes are automatically in BCNF in such |R, viz., those of entity sets, of purely "one-related" relationships, and of binary relationships. Here we give one condition sufficient for BCNF for all other schemes in |R, that the underlying regular ERD also be "loop-free".

In the third section we shall explain the "master-file approach" formally, and show that it enables one to test the BCNF-ness of the particular databases that can be defined to conform with it. This "restrictive design methodology" is then discussed in the last section.

B. ENTITY-RELATIONSHIP PRELIMINARIES

Until recently most authors have used E-R diagrams as concise and intuitive database design indicators, using entities and relationships as a "lingua franca" of data model theory. ERDs act as interfaces between the various conceptual models in the database literature. For example, in the 1982 ACM Symposium on Principles of Database Systems both the opponents [AtPa] and the proponent [Ulm82b] of the "universal relation view" used informal ERD's in their arguments.

We have seen logical analysis of the E-R model reach a new level, that of rigorous and careful treatment of many of the basic assumptions and definitions behind this "jacknife" of the trade [Ch76,80; JNS83a,b,c]. Not only do is this useful to help an understanding of the entity-relation model, but also the newer results have bearing on many issues of database theory. There is now an impetus to study its formalizable aspects with the same care that has been applied to the relational model. As a basic text in database systems [Ulm82a] explains, there is a natural representation of the E-R model in terms of either the relational or network or hierarchical data models.

The conversion of an ERD into a relational model, using one associated relation scheme for each entity or relationship, has become a standard, accepted method [Ulm82a]. We extend this known conversion one step farther, to ask: What do the given quantifying marks on the relation/entity connectors in the E-R diagram imply that we know about functional dependencies (FDs) in its canonical relational schema, of all the associated schemes? We feel that the proper interpretation of these marks, in direct accordance with standard mathematical thinking about many-to-one functions, leads inevitably to the definition of a "regular" ERD and of its "basic" (or fundamental) FDs, as seen below.

Example 0: Consider Figure 0, containing entity sets, EMPL and DEPT, and the relationship set EMPL_DEPT. These entity sets are converted into relation schemes, with the same names, as follows:

EMPL(E#,EN,JC); DEPT(D#,MGR,LOC).

Note that the respective keys are E# and D#, called the primary entity keys, determining the other attributes in their relation schemes, as seen by the quantifying mark 'n' on their connectors, and '1' on the dependent attributes. Hence, in the relationship scheme EMPL_DEPT we need not repeat the dependent attributes of these entities, but only give these keys plus any attributes directly dependent on the relation, here TASK and STart_DATE:

EMPL_DEPT(E#,D#,TASK,ST_DATE).

Note that the attributes EN,JC,MGR,LOC,TASK and ST_DATE are in fact all functionally determined by the key E# of EMPL, which is thus a key of the relation scheme EMPL_DEPT.

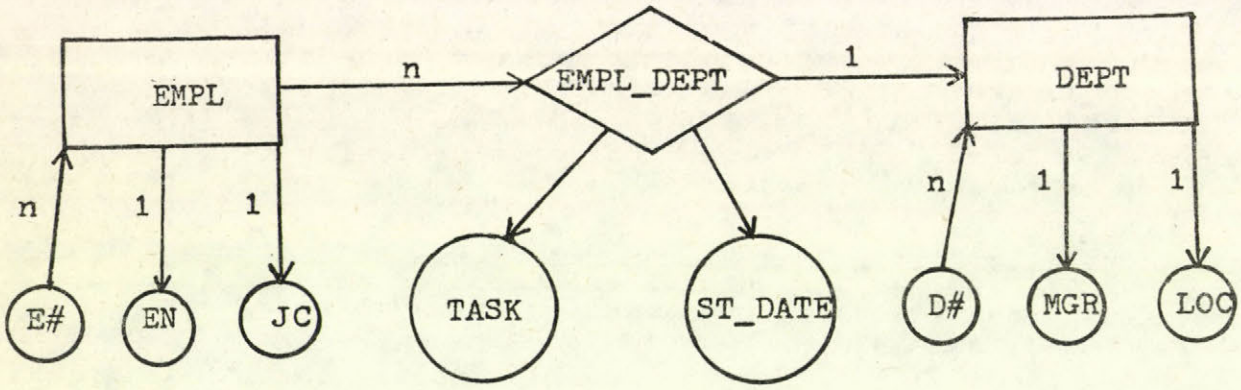


FIGURE 0. A REGULAR ERD

The above dependency analysis can be seen to be correct from the 'n' and '1' (partial) functional indicators on the arrows; i.e., we consistently mark all connectors in an ERD with indicators of the connected entity's functional participation in the affected relationship, whenever this information is known. Thus, by following the arrows, from n- to 1-related attributes, the ERD deducible functional dependencies (FDs) can be written down!

Two cautions are in order: a) to employ this convenience, we disallow any special types of attributes, e.g. those whose existence depends on other entities, which would need special designations; b) the device of "following the arrows" needs interpretation for relationships of three or more entities.

However, for normal entity schemes or for binary relationships as above our convention is clear. Thus, since the keys can be seen by inspection, it follows that the "basic FDs" of the ERD of Figure 0 are these:

E# -> EN, E# -> JC for the schema EMPL;

D# -> MGR, D# -> LOC for the schema DEPT; and

E# -> D#, E# -> TASK, E# -> ST_DATE for schema EMPL_DEPT.

Note that if we did NOT know the n/1 functionality in this last relationship, then we could only assume an arbitrary many-related indicator 'n' on both connectors between EMPL and DEPT, with the consequence being in that case: the only deducible key (from this information) would be (E#,D#). Also, if an k-ary relation has two or more entities (say P1 thru Ps) with 'n' indicators, and

all others marked '1', then the indicated key is P1....Ps. This makes sense if one thinks of the arrows as showing the many-one functionality from the several n-related arguments, Pi, to the other attributes, a unique tuple of values of which is determined by fixing an s-tuple of values for the key.

We shall call all relationship schemas with at least one n-related entity n-RELs: if they have any unmarked entities ('n', by default) we also call them n-RELs. In either case the indicated key would be the union of all the n-related entities' primary keys. In practice an "all n" case is rare. Less rare is the case of all entities being mutually determining keys, i.e. Ei -> Ej, all i and j. When this happens we call the relationship schema 1-REL, because all its Ei should be marked '1'.

RULES FOR THE BASIC FDs GIVEN BY A (REGULAR) E-R DIAGRAM:

For each entity set relation scheme ENT(PK, A1, ...,Ae), where PK is the pre-chosen primary key, we have these 'basic' FD's

RULE 1: F(ENT) = {PK -> Ak : 1<=k<=e.}

For each 1-REL relation scheme, the set of 'basic' FD's is

RULE 2: F(1-REL) = {Ei -> Aj : Ei 1-related, Aj any attribute}.

Finally, for any n-REL relation scheme, where P1,...,Ps are the primary keys of the n-related entities, the 'basic' FD's are

RULE 3: F(n-REL) = {P1...Ps -> Ai : Ai any other attribute}.

The "canonical relational (database) schema" of an ERD is the set of all its associated relation schemes. E.g., for Example O, it can be denoted !R = { EMPL, DEPT; EMPL_DEPT }.

DEFINITION: If an ERD has only relationship schemes of the types 1-REL and/or of the type n-REL, with the basic FDs for each of its entity and relationship schemes as defined above in Rules 1 - 3, then we call the ERD, and its canonical relational schema !R, <u>regular</u>.

NOTE: Since all these rule-given FD's are the only ones clearly visible from the ERD itself, as the database design, and because the ERD is presumed to have clear semantics, the set of all the basic FDs is assumed to be a cover for all (the known) FDs of !R. This assumption is implicit in the definition of "regular" ERD.

Clearly, the ERD of Figure O, with the FDs we have deduced from it as above, is a regular ERD.

DEFINITION: A relation schema R is said to be in Boyce-Codd Normal Form (BCNF) with respect to a set F of FDs if, for any FD X -> A embedded in R, either A is in X (i.e., this FD is trivial) or else X contains a key of R (with respect to the set F). We also say that a relational schema <IR, IF> is in BCNF, ("globally") where IR is a set of relation schemes <Ri, Fi> with FD sets Fi over Ri, if each Ri is in BCNF with respect to all the FDs derivable from IF = Ui Fi. We informally say that a regular ERD IE is in "BCNF" iff its canonical relational schema <IR, IF> is, where IF is the family of all its basic FDs.

As examples showing the range of such conditions, we exhibit an (upper conceptual domain of an) ERD which is not in BCNF and a larger diagram which is.

Example 1. In this ERD of an Employee/Department/Project relationship, each EMPL can work in more than one DEPT, and only the combination of DEPT and EMPL determines the Project now being worked on by that employee in that department. Each Project is wholly contained in a single Department. The ERD is NOT in BCNF, because PROJ -> DEPT is a violating FD.

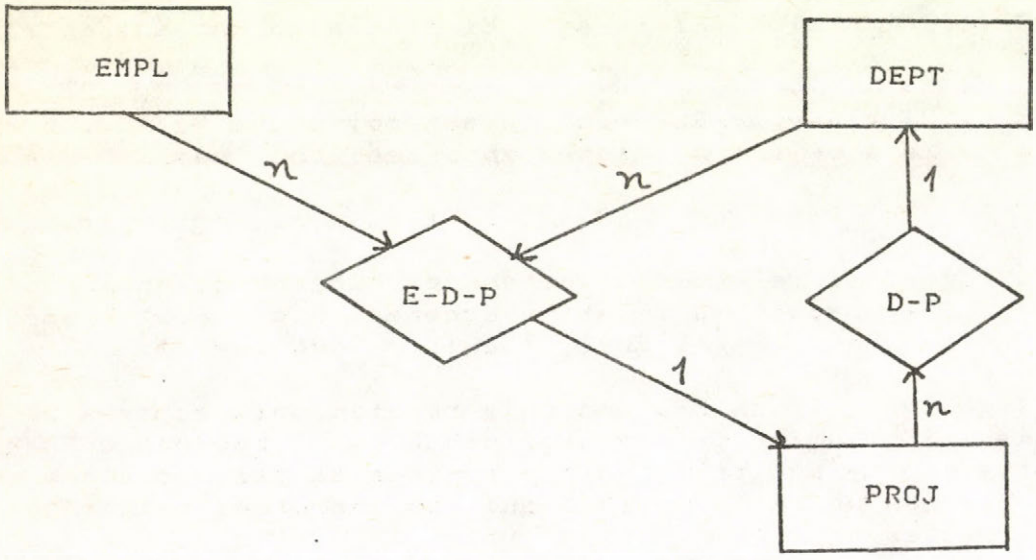


FIGURE 1. Regular ERD not in BCNF

Example 2. This is from a real-world case study of a business enterprise's accounting functions, of a complicated and very inter-related nature [Ulm82b]. (The sample subdiagram shown here represents less than a third of the original!) With applications of this complexity being common, some very realistic databases

can be seen to be in BCNF, as long as they are well structured. In this case the schema is in BCNF because all relationships are in fact binary; cf. Proposition 1 of Section II.

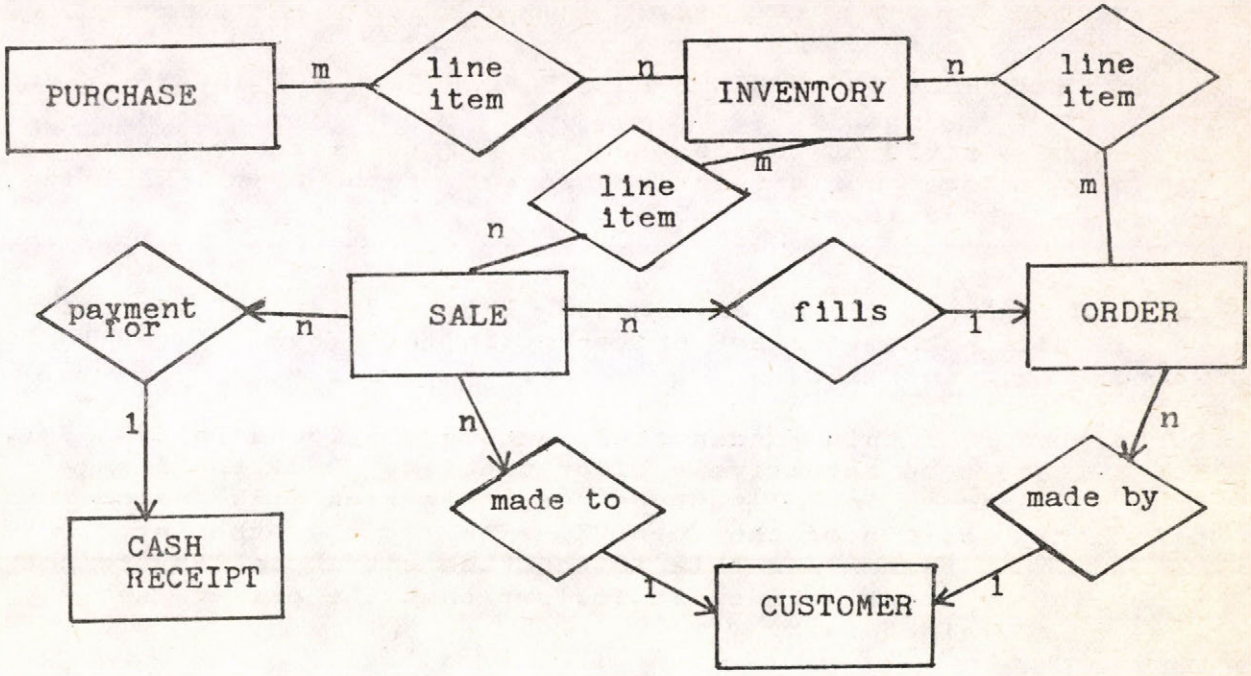


FIGURE 2. Enterprise's schema, in BCNF

II. SUFFICIENT CONDITIONS FOR BCNF

In this section we shall describe some general conditions that are sufficient to imply BCNF. This preface to our Section III discussion of the scenario is needed for two purposes: a) to help orient the reader to terminology used later, and b) to show the power of certain general conditions by exhibiting their desirable consequence when assumed as a "package", namely BCNF.

We assume that the reader is familiar with the principles of database dependencies, as found in [Ulm82a]. We shall adopt the notation and terminology there. Below we shall assume that |E is a regular ERD whose canonical relational scheme can be denoted

|R = (S1, S2, ..., Sn), and that these n relation schemes have their basic sets of FDs, as defined in Section I, denoted F1, F2, ..., Fn. Let the set of all basic FDs of |E be denoted

F(|E) = F1 U F2 U ...U Fn. By the nature of Rules 1-3, each FD K -> A in an Fi must enjoy these properties:

1. be embedded in an Si: KA $\subseteq$ Si;

2. have in K only primary key attributes of entity sets;

3. have a single attribute of Si as its right-hand side A;

4. have a key (or superkey) of Si as its left-hand side K.

A. MANY RELATION SCHEMES FOR REGULAR ERDs ARE ALWAYS IN BCNF

Indeed, the title of this subsection is true for any entity set relation scheme. For a relationship-set of the type 1-REL, its relation scheme is also in BCNF.

THEOREM 1. LET |E be any regular ERD, and let ENT represent the relation scheme of an entity-set in |E. Then ENT is in BCNF with respect to F(|E).

Formal proof of this fundamental fact can be found in [JNS83a]. But it should be intuitively clear that any FD of the form X -> A that is embedded in ENT must be derivable from F(ENT) alone, which only has FDs of the form PK -> Ak, Ak any other attribute of ENT. One reason for this is that the attributes Ak are not found in any other scheme. It follows that the primary key PK must be contained in X.

There is little syntactic distinction, at root, between ENT relation schemes with different, equivalent candidate keys and the relation schemes of type 1-REL. Often, the choice of entity-set or 1-related relationship to represent an "object" like Department, is arbitrary. This suggests that Theorem 1 may be extendable to relation schemes of 1-REL type. We find this to be the case.

THEOREM 2. Let REL be a relationship-set relation scheme, in a regular ERD |E, which has entity-sets E1, ..., Et all 1-related. Then REL is in BCNF with respect to F(|E).

Again, a formal proof can be found in [JNS 83a], based on the simple observation that if X -> A is in any nontrivial FD true in REL, then by regularity it must be derivable from F(REL). Thus, X must contain the primary key of some Ej, $1 <= j <= t$, any of which is a key of REL.

By exhaustive consideration of the cases of keys for any binary relationship R(A, B), we can also conclude the following proposition, which is the basis for the claim that Example 2, despite its complex inter-connections, in in BCNF. In the "worst case", where R is a many-to-many relation, even if some external

connections imply the FD A -> B, we simply conclude that the key A of R must be included on the left-side of any non-trivial FDs derivable from F(R). (Another FD B -> A means B is also a key.)

PROPOSITION 1: If REL is a binary relationship in a regular ERD |E, then REL is in BCNF with respect to F(|E).

B. LOOP-FREE ERDs HAVE ALL RELATION SCHEMAS IN BCNF

The relationship-set relation schema E_D_P (EMPL, DEPT, PROJ) of Example 1 is not in BCNF, because by Rule 3 applied to D_P(DEPT, PROJ), the FD PROJ -> DEPT becomes embedded in the schema E_D_P, where PROJ is NOT a key. We seek a condition to rule out such cases. Note that there is no way to decompose E_D_P into two binary relationships without losing some semantic information.

We conclude that the problem in Example 1 arises from the fact that this ERD contains a loop involving at least two distinct relationship sets. Below we define formally what we mean by "loop" and "loop-free" for an arbitrary regular ERD. Note that, as in this example, it suffices to consider only the entites and relationships, the "upper conceptual domain" of the ERD, rather than the low-level attributes, because the latter do not affect the existence of loops OR of BCNF-violating FDs in the ERD.

DEFINITION: Let the upper conceptual domain U(|E) of an ERD |E be identified as the hypergraph whose nodes are all the entity sets Ei of |E and whose (hyper)edges are all the relationship sets Rj of |E. Each Rj as an edge is the set of distinct entity sets related by Rj. (This notion of possibly non-binary edges generalizes the usual edge notion in graphs [Berg].)

Suppose E and E' denote entity sets in |E. A path from E to E' is a sequence of distinct relationships R1, ..., Rs such that:
E is in R1, E' is in Rs, and Ri $\cap$ Ri+1 $\neq \emptyset$, 1<=i<=s.
The length of the above path is s. A loop based at E is a path from E to E of length at least two. We shall call |E loop-free if there does not exist a loop based at any node in U(|E).

PROPOSITION 2: An ERD |E is loop-free if and only if there is no sequence in U(|E) of the form

(R1, E1, R2, E2, ..., Rs, Es, Rs+1), where

1. E1, E2, ..., Es are distinct entity sets;
2. R1, R2, ..., Rs are distinct relationships;
3. s is at least two, and Rs+1 = R1;
4. Ei is in (and so related by) both Ri and Ri+1, 1<=i<=s.

This Proposition shows that IE is loop-free if and only if its hypergraph is Berge-acyclic [Berg], equivalent here to 1-4, and its proof follows directly from the definitions.

THEOREM 3: Let IE be a regular ERD that is also loop-free. Then every relation scheme S in the canonical relational schema IR of IE is in BCNF with respect to F(IE).

[The formal proof of Theorem 3 is too long to include here, but is based on the intuitive idea that if no loops exist in the ERD, then no "externally implied" FDs can become embedded even in a non-binary, n-REL type relation scheme S. Notice that we already know, by Theorems 1 and 2 and Proposition 1, that all other types of relation schemes in IR are in BCNF with respect to F(IE).]

Example 3: The ERD in Figure 3 is the upper domain of the COMPUCO database (from the manual of a popular DBMS), and it is clearly loop-free. If we assume it is also regular, then we can write down the important, inter-entity FDs simply by following Rules 1-3. However, even before seeing the FDs, we know by Theorem 3 that none of them will violate the BCNF conditions for the relationship TRANSAX, which records daily transactions involving customers, vended products, and sales representatives of a value-added-reseller, where each product contains a certain type of computer. For example, we see that the key transactid of TRANSAX cannot be determined by any combination of the primary entity keys: empid, custid, prodid.

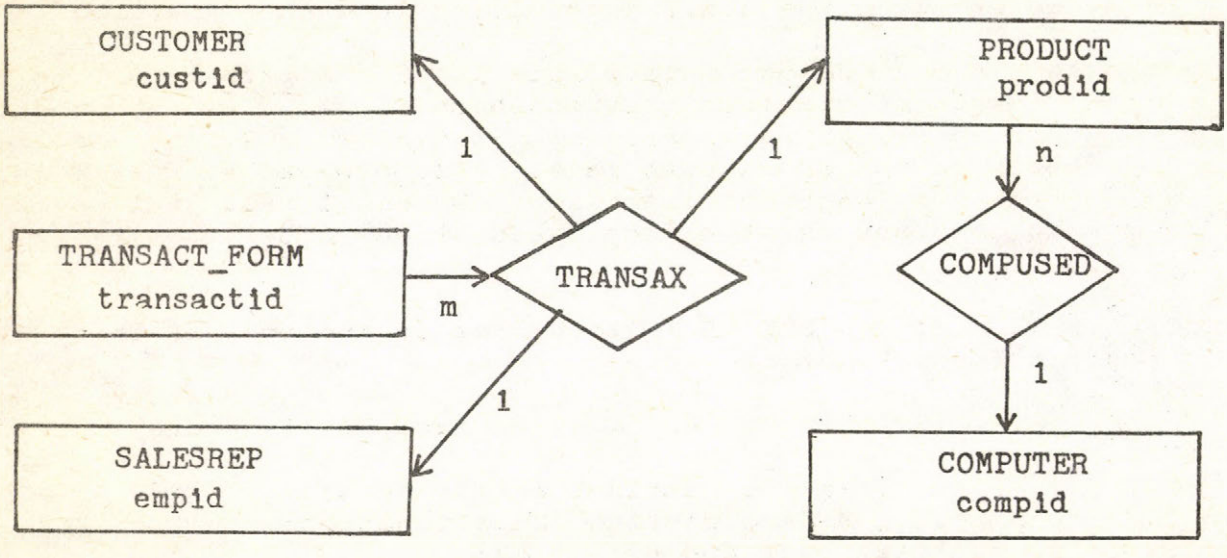


FIGURE 3. The upper ERD of the COMPUCO database.

NOTE: Although Theorem 3 can sometimes be used to conclude that an entire database schema will be in BCNF, even before all the FDs are explicitly seen, the loop-free condition is very much stronger than the BCNF condition. That is, the former is not a necessary condition for the latter, as the following example shows. Also, cf. Example 2, which is in BCNF by Proposition 1, but contains many loops.

Example 4: The entities are Contract, Supplier, Part, and pRice. The parts and their prices are dependent by bid on the contract, but not on the supplier. There is only one supplier for each contract, but a supplier can charge different prices on different contracts. The explicit keys are underlined:

$$\mathrm{IR} = [\ \underline{\mathrm{CS}}\mathrm{PR};\ \underline{\mathrm{CS}};\ \underline{\mathrm{CP}}:\ \underline{\mathrm{CR}};\ \underline{\mathrm{SR}}\ ].$$

Theorem 3 does not apply to this IR, because its easily diagrammed U(IE) has loops, but it is clearly in BCNF.

PROPOSITION 3: There are P-time algorithms to determine whether any ERD IE is: (1) regular, for a given set IF of FDs, or (2) loop-free.

Proof: One can test regularity, given the sets IF and F(IE) of FDs, by testing if $F(\mathrm{IE})^+ = \mathrm{IF}$ by use of the closure-membership algorithm [Ulm82b], in time linear in the product of the sizes of the sets. By Proposition 2, the loop-free test can be done by standard algorithms for cycle-checking in P-time; cf. [Fag81].[]

III. SPECIAL CONDITIONS MAKING BCNF P-TIME TESTABLE

A. THE MASTER-FILE SCENARIO FORMALIZED

Many modest-sized databases have regular ERDs with canonical relational schemas that exemplify the "unique masterfile" scenario, mentioned in the Introduction. For convenience we continue to study their ERD models here via the simplified canonical relational schema which ignores the lower conceptual domains' proper parts: dependent attributes of entity and relationship sets. That is, we consider only the entities - as basic units - and the relationships between them, the "upper conceptual domain", in order to focus on the open question of testing BCNF. We can do so without loss of generality because the BCNF conditions can only be violated in regular ERD's by certain functional dependencies between entities. i.e. between their primary key attributes. Thus, entities are here represented solely by primary keys, the only attributes kept.

For our real-world-oriented class of ERDs we can formalize the masterfile scenario as follows, assuming regularity. The class includes all ERDs having canonical relational schemas of form:

(#) $IR = (E1,\ldots,Em: R1,\ldots,Rn: S)$, where:

a. the Ei are all the entity set relation schemes;

b. S is the only relationship scheme which can relate one or more m-related entities, except for the binary relationships;

c. the Ri are the other relationships' schemes, either binary relationships (possibly n-to-1, n-to-n, or 1-to-1) or else "equivalence" relationships where all entities are 1-related: 1-RELs.

d. furthermore, the m-related entities of S, say P1,...,Ps, include a key or a determiner of a key for each entity set Ej, i.e., P1...Ps -> Ej, $1<=j<=m$. Thus, S contains a "database key" [AtPa].

While another type of Ri would be logically consistent, one that has only one n-related entity and two or more 1-related entities, for simplicity we assume this type decomposed into two or more binary relationships, without losing generality. This is a valid simplification when the ERD is regular. For example, the relationship R(ABC) with FD's A -> B and A -> C for A n-related and B,C 1-related. By standard methods, R(ABC) is decomposable into R'(AB) and R"(AC), losslessly, since A = AB ^ AC -> BC [Ulm82a].

B. TRANSFORMATION METHOD: UNIT FDs and KEY EQUIVALENCES

The masterfile scenario (#) desribed above can be subjected to the analysis of a transformational method that may reduce its set IK of key dependencies to a "linear" set (of unit FDs), whose full set of keys can be discovered in P-time. Since the methodic discovery of a new key, even for a single relation scheme, is the NP-complete problem at the source of the intractibility of BCNF [Osb], such a transformation would solve the BCNF problem in this case. To solve it in more general cases seems impossible by this special method, because the needed assumptions are rather closely fitted to our formal description of the given scenario.

The assumed scenario IR models a type of well-designed database: there is one masterfile, S, which contains at least a determiner of a key for each of the other files, the latter files having either one key or else several equivalent keys. Some well-designed, regular ERD's have canonical relational schemata that can be described by the scenario above: Examples 3 and 4 are such cases "in the small", but many others can be imagined, where

one file like the Transact_Form file determines a key of all the other files.

The main contribution of this section of the paper is chiefly the insight that a fairly obscure theorem, published by the Czech researcher J. Pokorny, applies to this subclass of the regular ERD's [Pok]. We use it to help test BCNF: to do so when normal forms are untreated in Pokorny's paper, takes some mathematical manipulation, but the "key" machinery is available in his work!

Application to this special scenario (#) will show the logical power of Pokorny's theorem, but it is impossible within the constraints here even to outline its long proof. Essentially, the result here can be applied to relational schemas other than as depicted in the scenario, as long as their non-binary relation schemes having m-related entities are losslessly joinable into a single scheme, S. The exact conditions under which such joining is possible is worthy of further research but is beyond the scope of this work, which only seeks to clarify the basic approach.

Concerning the masterfile S, the basic Rule 3 FD's of S are in:

$$F(S)=\{ P1...Ps \rightarrow Ej,\ 1<=j<=t\},$$

where the Pi are the m-related and the Ej the 1-related entities of S. [We may assume by renumbering that these entities are the first t in |R.] Of course, P1...Ps is not necessarily a minimal key of S, which is part of our problem. If we knew that P1...Pr, r<s, say were a minimal key then we would retain only P1,.....,Pr as the m-related entities in S and relegate the other Pj's to be among its 1-related entities. Since the minimal left-hand sides of the t FDs in F(S) can be determined easily, by standard algorithms, we shall assume that F(S) is minimal.

Another assumption needed for using Pokorny's Theorem is the non-redundancy of left-hand side attributes in the FDs of F(|R). But this is trivial for the unit FDs in the other schemes, and we have just guaranteed by the minimality of F(S) that we do not have either Pi->Pj or Pj->Pi.

Note that all the fundamental FD's in |F - F(S) are of the form K -> A where K and A are entity-set primary (key) attributes, and K is the key of a relation scheme other than S. FD's in this simple form, with both left and right sides a single attribute, are called "linear" by Pokorny; following common usage, we call them "unit FDs" [BB79].

DEFINITION (ASSOCIATED SET OF UNIT FDs): Let us denote by

$$UFD(|R) = [\ F(|R) - F(S)\ ] \cup \{ Pi \rightarrow Ej: \text{all } i<=s, j<=t\},$$

the "set of unit FDs associated with" F(IR). The set UFD(IR) cannot be expected to be FD-equivalent to F(IR), but, under certain further conditions, it may be "S-key-equivalent", meaning that the sets of keys of S with respect to either F(IR) or UFD(IR) would be identical. This is a desirable state: it would enable testing S for the BCNF-ness! Indeed, Pokorny gives such conditions that are a) easily P-time testable, b) necessary when BCNF does in fact hold, and c) in any case strong enough to make the determination of BCNF possible in P-time.

To employ Pokorny's theorem in our setting, we define the directed graph

G(IR) = <(E1,...,Em); UFD(IR) >, where each unit FD

Ei->Ej in UFD(IR) is considered as an edge joining node Ei to node Ej. Thus G(IR) is the directed graph with nodes all the entity sets in IR and edges the unit FD's associated with F(IR). Since IR can be assumed to be a connected ERD (via the database key), it is easy to see that the strongly connected components of G(IR), hereafter called components, determine what sets of attributes are the keys of S under the FD constraints in UFD(IR). By one other assumption, Pokorny's theorem shows us how to determine that the same sets are also the keys of S under the FD's in all of F(IR)! We shall explain that other assumption, which is easy to check in our scenario case.

Recall that any key of S is a key/determiner of any entity-set Ei. By a source component in G(IR) we shall mean, as in [Pok], a component whose entities (all equivalent under unit fd's) have in-degree zero: there are no unit FDs from other components of which they are the right-hand side. Clearly, any key of S must consist of attributes Pi that are in separate source components. All source components are easily computable in (small) polynomial time, using Tarjan's algorithm [AHU].

Let the source components of G(IR) be denoted as S1,...,Sk.

DEFINITION: Let f: A1.....An -> E be any FD in F(IR). Let C be any non-source component in G(IR). We say that f is <u>hierarchial for C</u> if: (a) E is in C, and (b) none of the Aj are in C, for 1<=j<=n.

THEOREM 4: For the defined UFD(IR), the masterfile S of IR with the assumptions above, and F(IR) the set of basic FDs of IR having the minimality assumptions made above:

the sets of keys of S with respect to either F(IR) or UFD(IR) are identical, if and only if:

(*) for each non-source component C in G(IR), there is at least one FD f in F(IR) that is hierarchial for C.

NOTE: This is a restatement and application of Pokorny's Theorem 2 to our scenario. When this theorem is applicable, it will be tractable to check the BCNF conditions for S, because finding keys with respect to a set of unit FDs is solvble in P-time [Pok].

Suppose that the condition (*) of Theorem 4 is true. Then, in particular for any non-source component C holding an Ej of S: there is at least one FD in F(S) of the form P1...Ps -> Ej such that Ej is in C and no Pi is in C, i.e. Ej->Pi is NOT in F(!R). Indeed, this is case, because, as we next show, for our special scenario, this desirable condition (*) is in fact true!

LEMMA: For relational schema !R of the special form (#), with the minimality of F(!R) assumed above, the condition (*) of Theorem 4 is always true.

PROOF: In the acenario case (#), recall, the minimal left-sides of basic FDs in F(S) contain Pi that can be found in source components of G(!R). Now, each non-source component C lies at the end of an incoming edge (a unit FD) from some other component B. Let E in C be the right-side of such a unit FD. By the nature of the two kinds of unit FDs in G(!R), associated with the basic FDs in F(!R), there are two main cases:

(1) B is a source component, and the unit FD can be taken to be P -> E. Then either this is a basic FD of F(!R) or else, when E is in S and P is some Pi, a minimal FD P1...Ps -> E is in F(!R). In either subcase, none of the left side P's is in C because C is non-source;

(2) B is a non-source component distinct from C, and the connecting unit FD in G(!R) is say D -> E, where D is in B. But this implies that D -> E is a basic FD in F(!R), and that D is not in C because distinct components are disjoint.

Thus, in either main case for C an arbitrary non-source component, there is an FD in F(!R) which is hierarchial for C.[]

COROLLARY. For a relational schema of a regular ERD, in the form
(#) !R = (E1,...,Em; R1,...,Rn; S), as described above, the property of being in BCNF can be decided algorithmically in polynomial time.

PROOF: Recall that all schemes in !R, except possibly S, are already in BCNF, by Theorems 1 and 2. Because Theorem 4's condition (*) is true for !R, it can be applied to verify whether S is in BCNF wrt F(!R) by finding the keys of S with respect to the unit FD's in UFD(!R). (Pokorny also showed that the key-finding problem for a set of unit FD's is solvable in P-time). Then, for any FD X -> A in the scope of S, one will know whether X contains a key of S wrt F(!R), and so whether S is in BCNF. Since the basic FDs of F(!R) form a cover of all FDs, the whole process can be done in time polynomial in the size of F(!R). []

IV. CONCLUDING REMARKS

We have seen two possible approaches to the problem of whether a relational database schema $\mathcal{R}$ is in BCNF, where we consider $\mathcal{R}$ the canonical schema of a regular ERD $\mathcal{E}$. In the first approach, that of finding sufficient conditions to imply BCNF, we saw incidentally that many relation schemes in $\mathcal{R}$ are in fact in BCNF with respect to $F(\mathcal{E})$ due to the regularity of $\mathcal{E}$: namely, the schemes of entity sets (Theorem 1), of any binary relationships (Proposition 1), and of any purely 1-related relationship sets (Theorem 2). To ensure that every relation scheme in $\mathcal{R}$ is in BCNF, we defined the loop-free condition for an ERD; this condition, in addition to regularity, obtains the desired result (Theorem 3). There are P-time algorithms to determine the regularity or loop-freeness of arbitrary ERDs (Proposition 3).

However, loop-freeness is equivalent to Berge acyclicity of $\mathcal{R}$, which is the strongest notion of acyclicity usually studied for relational schema [Fag81]. It is also a fairly restrictive condition: an ERD with two different relationships that share at least two entity sets (as in Example 1) violates it. Fagin's results on Berge acyclicity show it eliminates ambiguity in navigational paths to answer queries, and so it is nonetheless a desirable property, both from this aspect and from the fact that the implied BCNF condition prevents certain of the classical update and insertion anomalies [LePa].

For our second approach to the BCNF problem, we posited an ERD-based design scenario suggested by several examples that makes it possible to test for the condition. To see that this test can be done in the scenario setting, it was helpful to use the "key-equivalence via unit FDs" result of Pokorny (Theorem 4). In its corollary, we have given a method that determines truth or falsity of BCNF, under quite feasible conditions. These conditions include the non-existence of pairs of redundant attributes P1....Ps in the primary key of S, which holds a database key, implying all other attributes. To reduce the seeming "restriction" of this design requirement, note that the DBA can avoid redundancy by combining equivalent attributes into one entity, with separate fields for the attributes.

Overall, we have seen that when the Section III relational model $\mathcal{R}$ of the scenario is used as a guide, a specialized database design methodology is obtained that has the desired property: its specific cases, various relational schemata, can be tested for BCNF in polynomial time. The scenario design model also has other pleasing properties, e.g, that its most complex relation scheme S acts as a "masterfile" for the database, and that its own minimal keys are easy to find. While this model is clearly not universally applicable, in many cases (e.g., where a database key can be given) it may turn out to be useful, at least as a guide.

#####

REFERENCES

[AHU] Aho, A., Hopcroft, J. and Ullman, J. "The Design and Analysis of Computer Algorithms", Addison-Wesley, Reading, 1976

[AtPa] Atzeni, P. and Parker, D.S. Assumptions in relational database theory. Proc. 1st ACM Conf. Principles DB Systems, 1982

[BB79] Beeri, C. and Bernstein, P. Computational problems related to the design of normal form relational databases. ACM Trans. Database Sys., 4(1), Mar. 1979

[Berg] Berge, C. "Graphs and Hypergraphs", North-Holland, Amsterdam, 1976.

[Ch76] Chen, P. The entity-relationship model: towards a unified view of data. ACM Trans. Database Sys., 1(1), Mar. 1976

[Ch80] Chen, P. (ed.) "Entity-Relationship Approach to Systems Analysis and Design", North-Holland, Amsterdam, 1980

[Codd] Codd, E.F. Recent investigations in relational database systems, Proc. 1974 IFIP Cong., North-Holland, 1974

[Fag81] Fagin, R. Types of acyclicity for hypergraphs and relational databases, Technical Report RJ 3330, IBM San Jose, 1981

[JF] Jou, J. and Fischer, P. The complexity of recognizing third-normal form, Info. Proc. Letters 14 (4), June 1982

[JNS83a] Jajodia,S., Ng,P. and Springsteel, F. Entity-relationship diagrams which are in BCNF, Intl. J. Comp. Info. Sci. 12 (4), 1983

[JNS83b] Jajodia, Ng, and Springsteel. Problems of equivalence for entity-relationship diagrams, IEEE Trans. Software Engr. 9, 1983

[JNS83c] Jajodia, Ng, and Springsteel. On universal and representative instances for inconsistent databases. Proc. 3rd Entity-Relationship Conf., North-Holland, 1983

[LePa] LeDoux, C. and Parker, D.S. Reflections on Boyce-Codd Normal Form, Proc. 8th Intl. Conf. Very Large Data Bases, 1982

[Lien] Lien, E. On equivalence of database models, J.ACM 29(2), 1982

[Osb] Osborn, S. Testing for existence of a covering Boyce-Codd Normal Form, Information Processing Letters 8 (1), 1979

[Pok] Pokorny, J. Key-equivalence of functional dependency systems, Proc. 10th Symp. Math. Foundations Computer Science. Lecture Notes in Comp. Sci. #118, Springer-Verlag (1981)

[Ulm82a] Ullman, J. "Principles of Database Systems", Second Edtn., Computer Science Press, Rockville, Maryland, 1982

[Ulm82] Ullman, J. The U(niversal) R(elation) strikes back! Proc. 1st ACM Conf. Principles Database Systems, 1982

Prof. F.N. Springsteel

University of Missouri at Columbia
Department of Computer Science
Columbia, Missouri 65211 USA
Telephone: (314) 882-4480, 882-3842

A restricted design methodology to allow testing for BCNF in polynomial time

F.N. Springsteel

Summary

In a relational database it is important to be able to test a proposed database design for the Boyce-Codd Normal Form (BCNF) condition. In the paper conditions are given that are either sufficient for BCNF or show that BCNF cannot be tested in polynomial time. There are two sets of conditions. The first one logically imply BCNF and it provide a setting which helps to suggest the second one in which it is easier to test for BCNF. The proposed methodology works for many databases expressible by E-R diagrams which have a "database key".

Egy korlátozott tervezési metodika BCNA tesztelésére polinomiális időben

F.N. Springsteel

Összefoglaló

A relációs adatbázis modell esetén igen fontos kitesztelni a javasolt adatbázis tervet, vajon teljesiti-e a Boyce-Codd-féle Normál Alak /BCNA/ feltételt. A cikkben olyan feltételek vannak megadva, amelyek vagy elégségesek a BCNA-hoz, vagy megmutatják, hogy a BCNA-t nem lehet polinom idő alatt kitesztelni. Kétfajta feltételrendszer van. Az elsőnek logikai következménye a BCNA és egyben megmutatja hogyan néz ki a második, amelyben a BCNA-t már könnyebben lehet tesztelni. A javasolt metodika különösen alkalmas azokra az adatbázisokra, amelyeket "adatbázis kulcs"-al rendelkező E-R diagramm segitségével lehet leirni.

MTA SZTAKI Közlemények 37/1987 pp. 171-183

DESIGN TOOLS FOR LARGE RELATIONAL DATABASE SYSTEMS

B. THALHEIM

Technische Universität Dresden
Sektion Mathematik
GDR 8027 Dresden

INTRODUCTION

Today, database is a fashionable word. Two opposing research directions in databases were initiated in the early 70's, the introduction of the relational model and the development of semantic database models. The relational model revolutionized the field by dramatically separating data representation from underlying implementation. Significantly, the inherent simplicity in the relational model permitted the development of powerful, nonprozedural query languages and many useful theoretical results. Much effort is being devoted to establish a concrete foundation for database technology in order to design more efficient and transparent systems. The first and almost solved foundation step for database technology is the precise definition of data model. A database model is a collection of mathematically sound concepts defining the desired structural and behavioural properties of objects involved in a database application. In the axiomatic approach, database models are defined by the properties of its structures and operators using conventional mathematics and logic to define the structural and behavioural properties of objects within the database model. The second foundation step for database

This work was partially supported by the Hungarian National Scince Foundation, Grant No. 1066/ 1985

technology is the precise definition of semantics of data model. In terms of logic, the semantics of each database within the database model can be deduced precisely by the application of valid inference rules to sets of axioms. The semantics of a syntactically correct schema is given by the axioms which characterize the databases to be accepted, the "real world" databases.
One of the most important database models is the relational model. All data are seen as being stored in tables, with each row in the table having the same format. Each row in the table summarizes properties of some object or relationship in the real world. One benefit and aim of the relational model is to provide a methdological approach for design of schemata and databases. This benefit is based on a powerful theory whose kernel is the theory of dependencies and constraints. Database dependencies can be seen as a language for specifying the semantics of databases. They constitute an inherent property of database systems and express the different ways that data are associated with one another. Today we know at least five arreas of application of dependency theory: normalization for a more efficient storage, serch and modification; reduction of relations to subsets with the same information together with semantic constraints; utilization of dependencies for deriving new relations from basic relations in the view concept or in so-called deductive databases; verification of dependencies for a more powerful and user-friendly, nearly natural language design of databases; transformation of queries into more efficient search strategies.

In this paper some tools for database design with database dependencies are considered. More user-friendly functional dependencies are introduced in chapter 1. In chapter 2, the utilization of negative information is examined. A reduction principle is introduced in chapter 3. The horizontal decomposition with union constraints is analyzed in chapter 4.

Now the main basic conceptions are briefly introduced.

A (strong many-sorted relational database) preschema $S = (U , Rel , \preceq)$ is given by a finite set U of attributes, by a finite set $Rel = \{P_1, \ldots, P_k\}$ of predicates or predicate

names and by an arity function τ : Rel $\longrightarrow U^+$ which associates with every predicate P from Rel a string $\tau(P) = A_1 \ldots A_n$ of U . Let $V_U = (V_A / A \in U)$ be a family of variables. The set $L(S,V_U)$ of (S,V_U)-formulas is the smallest set whose elements are strings from $V_U \cup$ Rel meeting the following conditions:

(1) If $P \in$ Rel with $\tau(P) = A_1 \ldots A_k$ and $x_i \in V_{A_i}$ $(1 \leq i \leq k)$ then $P(x_1,\ldots,x_k) \in L(S,V_U)$ (or briefly $P(\tilde{x})$ or $P(\tilde{x},\tilde{y})$).

(2) If $x,y \in V_A$ then $x=y \in L(S,V_U)$.

(3) If α, $\beta \in L(S,V_U)$ and $x \in V_A$ for some $A \in U$ then $\{(\alpha \wedge \beta), (\neg\alpha), (\alpha \rightarrow \beta), \forall x \alpha, \exists x \alpha\} \subseteq L(S,V_U)$.

A relational database <u>schema</u> is a pair $S^* = (S,\Sigma)$ of a preschema S and a set of formulas Σ from $L(S,V_U)$ called integrity constraints.

For a given preschema $S = (U, Rel=\{P_1,\ldots,P_k\}, \tau)$ a S-database $M = (\underline{D}_U , R_1,\ldots,R_k)$ is given by the family $\underline{D}_U = (D_A / A \in U)$ of domains and by an Rel-indexed family $(R_1,\ldots,R_k)$ of relations, so that every predicate P_i Rel with $\tau(P_i) = A_1 \ldots A_m$ the relation R_i is a non-empty finite set of functions r from $\{A_1,\ldots,A_m\} = /\tau(P_i)/$ to $\bigcup_{j=1}^{m} D_{A_j}$ with $r(A_i) \in D_{A_i}$. We denote the function r by an m-tuple $(r(A_1),r(A_2),\ldots,r(A_m))$.

Now interpretations I_M of V_U on S-databases M can be introduced as mappings I_M: $V_U \longrightarrow \underline{D}_U$ with $I_M(x) \in D_A$ for $x \in V_A$. In the usual way, the (M,I_M)-satisfaction of formulas α , denoted by $\models_M \alpha[I_M]$, can be defined. A formula $\alpha \in L(S,V_U)$ is said to be valid in M , denoted $\models_M \alpha$, if $\models_M \alpha[I_M]$ for each interpretation I_M of V_U on M . M is said to be a model of a set Σ of formulas if any $\alpha \in \Sigma$ is valid in M .

A set of formulas Σ implies a formula α if for any model M

of Σ $\models_M \alpha$ (denoted by $\Sigma \models \alpha$) .

Let $S^* = (S, \Sigma)$ be a schema. A S-database is called S^*-database if M is a model of Σ .

A (Hilbert-style) formal system Γ for a set K of formulas consists of axioms $\subseteq K$ and inference rules $\frac{\Sigma}{\alpha}$ ($\Sigma \subseteq K$, $\alpha \in K$). A rule $\frac{\Sigma}{\alpha}$ is called sound if $\Sigma \models \alpha$. A derivation of a formula α from a set Σ is a sequence of formulas $\alpha_1, \ldots, \alpha_k = \alpha$ where each α_i is an axiom, or a member of Σ , or is inferred from preceding formulas in the sequence by an inference rule of Γ . If there is a derivation of α from Σ in Γ we write $\Sigma \vdash_\Gamma \alpha$. A formal system Γ is called sound iff from $\Sigma \vdash_\Gamma \alpha$ follows $\Sigma \models \alpha$ and is called complete for a class K iff for $\Sigma \subseteq K$, $\alpha \in K$ from $\Sigma \models \alpha$ follows $\Sigma \vdash_\Gamma \alpha$.

1. FUNCTIONAL DEPENDENCIES

One of the directions of further development of relational database theory and relational databases is the improvement of friendliness of user languages. Most of languages proposed only idiosyncratic versions of operations of the relational algebra or calculi and "adhocness" prevailed over some clearly understood notion of simplicity.

Generalized functional constraints are formulas from $L(S, V_U)$ with $S = (\{A_1, \ldots, A_n\}, \{P\}, \tau)$, $\tau(P) = A_1 \ldots A_n$

$\forall y_1 \ldots \forall y_m (P(\vec{x}) \wedge P(\vec{y}) \wedge \alpha \longrightarrow \beta)$ where α and β are quantifier-free, predicate-free formulas, the set of variables occuring in $P(\vec{x})$ and $P(\vec{y})$ is exactly the set $\{y_1, \ldots, y_m\}$ and a superset of the set of variables occuring in α and β .

They are called in (SDPF 81) Boolean dependencies and considered in (THAL 85).

Example 1. We consider $S=(U=\{$LECTURER, COURSE'UNIT, GROUP, ROOM, TIME$\}, \{$TIMETABLE$\}, \tau)$ with some restrictions, for instance:

1) Any room is reserved only for one group at the same time.
2) If there is a lecture given by more than one lecturer then participants are different.

These constraints can be formulated with the following formulas:

$$\alpha_1 = \forall x_1 \ldots \forall x'_5 (P(x_1,x_2,x_3,x_4,x_5) \wedge P(x'_1,x'_2,x'_3,x'_4,x'_5) \wedge x_4 = x'_4 \longrightarrow (x_3 = x'_3 \vee (\neg x_5 = x'_5)))$$

$$\alpha_2 = \forall x_1 \ldots \forall x'_5 (P(x_1,x_2,x_3,x_4,x_5) \wedge P(x'_1,x'_2,x'_3,x'_4,x'_5) \wedge x_2 = x'_2 (\neg x_1 = x'_1) \longrightarrow (\neg x_3 = x'_3)))$$

Generalized constraints are of importance for a more natural definition of dependencies of functional kind and unifies all these dependencies. They can be introduced in a more intuitive manner.

By T_1^i the class of all n-ary Boolean functions $f(x_1,\ldots,x_n)$ with $f(1,\ldots,1) = i$ for $i \in \{0,1\}$ is denoted.

A pair (f,g) of n-ary functions from $(T_1^1 \times T_1^1) \cup (T_1^0 \times T_1^0)$ is called <u>generalized functional dependency</u>.

Given a S-Database $M=(\underline{D}_U,R)$, $U=\{A_1,\ldots,A_n\}$. For a Boolean function f we can define a binary relation $\tilde{f}$ on R:

$r \tilde{f} r'$ iff $f(\sigma'_1(r,r'),\ldots,\sigma'_n(r,r')) = 1$

where $\sigma'_i(r,r')$ denotes the function

$$\sigma'_i(r,r') = \begin{cases} 0 & \text{if } r(A_i) \neq r'(A_i) \\ 1 & \text{if } r(A_i) = r'(A_i) \end{cases} \quad (1 \leq i \leq n).$$

Now we define $\models_M (f,g)$ for pairs of n-ary Boolean functions:

$\models_M (f,g)$ iff for any $r,r' \in R$ from $r \tilde{f} r'$ follows $r \tilde{g} r'$.

Corollary 1. If for a pair of n-ary Boolean functions f,g and a S-database $M = (\underline{D}_U,R)$ $\models_M (f,g)$ holds then (f,g) is a generalized functional dependency. For any generalized functional con-

straint $\propto$ which has a model there exists a generalized functional dependency (f,g) such that for any S-Database M $\models_M \propto$ iff $\models_M (f,g)$.

Example 1 (continued). There are Boolean functions $f_1(x_1,\ldots,x_5)$, $f_2(x_1,\ldots,x_5)$, $g_1(x_1,\ldots,x_5)$, $g_2(x_1,\ldots,x_5)$ such that (f_1,g_1) and (f_2,g_2) are equivalent to $_1$ and $_2$:
$f_1 = x_4$, $g_1 = x_3 \vee \bar{x}_5$, $f_2 = x_2 \wedge \bar{x}_1$, $g_2 = \bar{x}_3$.

Some special generalized functional dependencies are the strong functional dependency, dual functional dependency, weak functional dependency (DEGY 82), functional dependency (DEAB 85), monotonic functional dependency (THAL 87) and key dependency. The theory of all these special generalized functional dependencies can be unified and simplified by a theory of generalized functional dependencies which is based on the following main characterization theorem for implication.
For Boolean functions f,g the inequality $f \leq g$ holds if for any value $\tilde{\sigma}$ from $f(\tilde{\sigma}) = 1$ follows $g(\tilde{\sigma}) = 1$.

Theorem 2 (THAL 85). Let $(f_1,g_1),\ldots,(f_m,g_m)$, (f,g) be generalized functional dependencies. Then $\{(f_1,g_1),\ldots,(f_m,g_m)\} \models (f,g)$ holds iff $\bigwedge_{i=1}^{m} (f_i \rightarrow g_i) \leq f \rightarrow g$ holds.

Corollary 3. Let be (f_1,g_1), (f_2,g_2) generalized functional dependencies.

1. If $f_1 \geq f_2$ and $g_1 \leq g_2$ then $\{(f_1,g_1)\} \models (f_2,g_2)$.
2. $\{(f_1,g_1), (f_2,g_2)\} \models (f_1 \wedge f_2, g_1 \wedge g_2)$.
3. $\{(f_1,g_1), (f_2,g_2)\} \models (f_1 \vee f_2, g_1 \vee g_2)$.
4. If $g_1 \leq f_2$ then $\{(f_1,g_1),(f_2,g_2)\} \models (f_1,g_2)$.

Example 1 (continued). With theorem 2 the following generalized functional dependencies follows from $(f_1,g_1),(f_2,g_2)$:
$(x_4 \wedge x_5, x_3)$, $(x_2 \wedge x_3, x_1)$, $(\bar{x}_1 \wedge x_2 \wedge x_4, \bar{x}_3 \wedge \bar{x}_5)$, $(x_2 \wedge x_4 \wedge x_5, x_1 \wedge x_3)$.

Corollary 4. For any system Σ of generalized functional depen-

dencies there exists an equivalent functional dependency (f_{Σ}, g_{Σ}).

Corollary 5. Testing whether two sets of generalized functional dependencies are equivalent is NP-complete. Testing whether two sets of generalized functional dependencies imply the same set of key dependencies is NP-complete.

From theorem 2 the known axiomatizations of different classes of special generalized functional dependencies can be derived. Several kinds of covers for functional dependencies can be better studied and understood using generalized functional dependencies.

2. EXCLUDED FUNCTIONAL DEPENDENCIES

It is useful, for database logical design, normalization and effective algorithms, to utilize the full information on given relations. It is well known that functional dependencies are the favourite constraints used to decompose relation schemes. This privilege is certainly due to the simplicity of the concept of functional dependencies, and to their widespread appearance in the real world. However, in a great number of applications there is a requirement to allow violation of some functional dependencies, i.e. functional dependencies that are desired, but that do not hold in the relation. For instance if there is given a database $M = (\underline{D}_U, R)$ and it is of interest which formulas from a class of formulas holds in M the strategy of recognizing the validity of formulas will be faster if negative information is used in next steps.

The formula

$\forall\tilde{x}\forall\tilde{y}\forall\tilde{y}'\forall\tilde{z}\forall\tilde{z}'\ (P(\tilde{x},\tilde{y},\tilde{z}) \wedge P(\tilde{x},\tilde{y}',\tilde{z}') \longrightarrow \tilde{y} = \tilde{y}')$ is called functional dependency and for $\tilde{x} = x_{i_1},\ldots,x_{i_k}$, $\tilde{y}=x_{j_1},..,x_{j_l}$, $X = \{A_i \,/\, i \in \{i_1,\ldots,i_k\}\}$, $Y = \{A_j \,/\, j \in \{j_1,\ldots,j_l\}\}$ denoted by $X \longrightarrow Y$.

The formula

$\exists\tilde{x}\exists\tilde{y}\exists\tilde{y}'\exists\tilde{z}\exists\tilde{z}'\ (P(\tilde{x},\tilde{y},\tilde{z}) \wedge P(\tilde{x},\tilde{y}',\tilde{z}') \wedge (\neg(y = y')))$ is called

excluded functional dependency and for $\tilde{x} = x_{i_1},\ldots,x_{i_k}$, $\tilde{y} = x_{j_1},\ldots,x_{j_1}$, $X = \{A_i / i \in \{i_1,\ldots,i_k\}\}$, $Y = \{A_j / j \in \{j_1,..,j_1\}\}$ denoted by $X \not\rightarrow Y$.

It is easy to find sound inference rules for functional dependencies and excluded functional dependencies.

(1) $\dfrac{X \rightarrow Y \,,\; X \rightarrow Z}{X \rightarrow Y \cup Z}$ (2) $\dfrac{X \rightarrow Y \,,\; Y \rightarrow Z}{X \rightarrow Z}$ (3) $\dfrac{X \rightarrow Y \cup Z}{X \cup W \rightarrow Y}$

(4) $\dfrac{X \rightarrow Y \,,\; Y \rightarrow Z}{X \cup V \cup W \rightarrow Z \cup W}$ (5) $\dfrac{X \rightarrow Y \,,\; X \not\rightarrow Z}{X \cup Y \not\rightarrow Z}$

(6) $\dfrac{Y \rightarrow Z \,,\; X \not\rightarrow Z}{X \not\rightarrow Y}$ (7) $\dfrac{X \cup V \not\rightarrow Y}{X \not\rightarrow Y}$ (8) $\dfrac{X \not\rightarrow Y}{X \not\rightarrow Y \cup Z}$ if $Y \neq \emptyset$

(9) $\dfrac{X \rightarrow Y \,,\; X \cup V \cup W \not\rightarrow Z \cup W}{Y \not\rightarrow Z}$ (10) $\dfrac{X \rightarrow Y \,,\; X \not\rightarrow Y \cup Z}{X \not\rightarrow Z}$

(11) $\dfrac{Y \rightarrow Z \,,\; X \cup V \cup W \not\rightarrow Z \cup W}{X \not\rightarrow Y}$ if $Z \neq \emptyset$.

The formal system Γ_1 consists of the axiom $\{X \rightarrow X / X \subseteq U\}$ and the rules (4), (9) and (11).

Theorem 6. The formal system Γ_1 is sound and complete for the class of functional dependencies and excluded functional dependencies.

The theorem is based on the following property and theorem 2.

Lemma. Let $\Sigma \cup \{\alpha\}$ be a set of functional and excluded functional dependencies. Then $\Sigma \not\models \alpha$ iff there is a database $M = (\underline{D}_U, R)$ with $/R/ = 2$, $\models_M$ and $\not\models_M \alpha$.

3. DEDUCTIVE BASIS OF RELATIONS

The normalization of relations is one of the most important tools for database design. The concept of special kinds of dependencies

has been proved to be useful in the design and analysis of relational databases. Besides the normalization there is possible another "normalization" of relations, the reduction of relations to a necessary basis. This basis will be called deductive basis since using special formulas we get the entry relation from its deductive basis. In most cases, the deductive basis is more efficient than other known normal forms. During the query phase, the formulas are used to generate all possible derivations of facts and thereby make them again explicit in the database.

Let $S = (U,\{P\},\tau)$ be a preschema with $U = \{A_1,\ldots,A_n\}$ and let $L(S,V_U)$ be a set of formulas.

A formula from $L(S,V_U)$

$\forall x_{11}\ldots\forall x_{mn}(P(x_{11},..,x_{1n})\wedge\ldots\wedge P(x_{m1},..,x_{mn}) \rightarrow P(x_{01},..,x_{0n}))$

is called <u>template dependency</u> if $x_{0i} \in \{x_{1i},x_{2i},\ldots,x_{mi}\}$ for any i, $1 \leq i \leq n$, and is called <u>join dependency</u> if moreover for all i,j $(1 \leq i < j \leq m)$ and k $(1 \leq k \leq n)$ from $x_{ik} = x_{jk}$ follows $x_{ik} = x_{0k}$ (no hidden conditions in premise).

Given a set Σ of template dependencies, a S-database $M = (\underline{D}_U,R)$ and the set of interpretations $I(M)$ of V_U on M.

Then we define the Σ-closure of M:

$Cl_0(\Sigma,R) = R$,

$Cl_{k+1}(\Sigma,R) = \{ r \mid$ there is an $\alpha = \forall\tilde{y}(P(\tilde{x}_1)\wedge\ldots\wedge P(\tilde{x}_m)\rightarrow P(\tilde{x}_0))$ in Σ and $I \in I(M) : I(x_i) \in Cl_k(\Sigma,R)$ $(1 \leq i \leq m)$ and $I(\tilde{x}_0) = r \}$ $\cup\ Cl_k(\Sigma,R)$;

$CL(\Sigma,R) = \bigcup_{k=0}^{\infty} Cl_k(\Sigma,R)$;

$CL(\Sigma,M) = (\underline{D}_U, CL(\Sigma,R))$.

<u>Corollary 7.</u> Given a S-database $(\underline{D}_U,R)$ and a set of template dependencies Σ, there is a finite natural number i such that $CL(\Sigma,R) = Cl_i(\Sigma,R)$.

Corollary 8. For any S-database M and a set of template dependencies Σ $\quad \models_{CL(\Sigma,M)} \Sigma$.

Given a set of template dependencies Σ and a S-database $M = CL(\Sigma,M)$. A subset R' of R is called Σ-deductive subset if $R = CL(\Sigma,R')$. A Σ-deductive subset R' of R which is minimal is called Σ-deductive basis of M .

There are two main problems.

1) Given a Σ-deductive basis R of M . How many steps are needed for the construction of the set $CL(\Sigma,R)$? Given a set Σ . Are there limits for the construction of the Σ-closure of databases ?

2) Given R and Σ . How to construct a Σ-deductive basis of R ?

For the second problem there are some algorithms. The first problem is harder. If for a given Σ the construction of Σ-closure of databases is unlimited then the utilization of Σ-deductive bases is not effective. In (THAL 84) the following property is proven.

Corollary 9. There are two binary join dependencies α_1, α_2 . and a template dependency α_3 such that for any natural i a S-database $M_i = (\underline{D}_U, R_i)$ exists with

$Cl_i(\{\alpha_1, \alpha_2\}, R_i) \neq CL(\{\alpha_1, \alpha_2\}, R_i)$ and

$Cl_i(\{\alpha_3\}, R_i) \neq CL(\{\alpha_3\}, R_i)$.

A set Σ of join dependencies is called Sheffer-set if there is an equivalent join dependency α_Σ .

Theorem 10. Given a Sheffer-set Σ of join dependencies. For any S-database $M = (\underline{D}_U, R)$ it holds that

$CL(\Sigma,M) = (\underline{D}_U, Cl_1(\Sigma,R))$.

4. HORIZONTAL DECOMPOSITION WITH UNION CONSTRAINTS

The purpose of this chapter is to introduce the notion of union constraints which is a type of database constraints not previously discussed in literature and to show that there exists a sound and complete formal system. In the database literature, there is a number of results, both positive and negative, for the existence of finite formal systems. The class of union constraints is the first class of nodependencies which is known to be axiomatizable. By an union constraint it is stated that there exists a cover of the relation with possibilities of "forgetting" some attributes.

We are given a preschema $S = (U,\{P\},\tau)$ with $U = \{A_1,\ldots,A_n\}$, $\tau(P) = A_1\ldots A_n$ and $X,Y \subseteq U$, $X \cup Y = U$.

The pair $[X,Y]$ is called <u>union constraint</u>.

A S-database $M = (\underline{D}_U,R)$ satisfies this constraint if there are subsets R_1,R_2 of R such that $R_1 \cup R_2 = R$ and $R = R_1(X) + R_2(Y)$ (denoted by $\models_M [X,Y]$) where $R'(Z)$ is the projection of R' on Z and $R' + R''$ is the generalized union on $\underline{D}_U$ or the sum (DEAB 85).

Obviously, the constraint $[X \cup Z, Y \cup Z]$ can be described with the following formulas from $L(S,V_U)$ for disjoint X,Y,Z :

$$\forall\tilde{x}\forall\tilde{y}\forall\tilde{z}\forall\tilde{x}'\forall\tilde{y}'\,(\,P(\tilde{x},\tilde{y},\tilde{z}) \rightarrow P(\tilde{x},\tilde{y}',\tilde{z}) \vee P(\tilde{x}',\tilde{y},\tilde{z}))\,.$$

Now we can define the formal system Γ_2.

Formal system Γ_2.

Ax $\quad [U,U]$

RU (1) $\dfrac{[X,Y]}{[V,W]}$ if $X \subseteq V$, $Y \subseteq W$ or $X \subseteq W$, $Y \subseteq V$

(2) $\dfrac{[X_1,X_2],\ [Y_1,Y_2]}{[X_1 \cap Y_1, Y_2]}$ if $X_1 \cap X_2 \subseteq Y_1$, $X_2 \subseteq Y_2$.

Theorem 11. The system Γ_2 is sound and complete for the class of union constraints.

The proof is this theorem uses an equivalence between union constraints and binary join dependencies.

REFERENCES

DEAB 85 C. Delobel, M. Adiba, Relational database systems. North-Holland, Amsterdam 1985.

DEGY 82 J. Demetrovics, Gy. Gyepesi, Logical dependencies in relational databases. MTZSAKI Tanulmanyok 133, 1982, Budapest, 59 - 78.

SDPF 81 Y. Sagiv, C. Delobel, D.S. Parker, R. Fagin, An equivalence between relational data base dependencies and a subclass of propositional logic. Journal of the ACM, 1981, 3, 435-453.

THAL 85 B. Thalheim, Funktionale Abhängigkeiten in relationalen Datenstrukturen. EIK, 1985, 1/2, 23 -33.

THAL 87 B. Thalheim, Dependencies in relational databases. Teubner, Leipzig, 1987.

Design tools for large relational database systems

B. Thalheim

Summary

In the paper some tools for database design with database dependencies are considered.
More user-friendly functional dependencies are introduced. The utilization of negative information is examined. Besides the usual normalization another "normalization " of relations, the reduction of relations to a necessary basis is introduced. Finally, the horizontal decomposition with union constraints is analyzed.

Nagyméretü relációs adatbázis-rendszerek tervezésének eszközei

B. Thalheim

Összefoglaló

A szerző a különböző függőségi tipusokkal biró adatbázisok tervezéséhez szolgáló eszközöket vizsgálja. Bevezet ujfajta "felhasználó-barátságosabb" funkcionális függőségeket. Megvizsgálja a negativ információ hasznositásának kérdéseit. Bevezet egy uj redukciós elvet. Végül analizálja az egyesités feltételeknél a horizontális dekompoziciót.

MTA SZTAKI Közlemények 37/1987 pp. 185-200

SEARCHING AND RETRIEVAL IN DATABASES BY TREES

H. THIELE

Department of Mathematics
Humboldt-University at Berlin
POB 1297, Berlin, 1086, GDR

1. Introduction

The present paper deals with a logical approach to searching and retrieval in (relational) databases by trees. We start with a many-sorted first-order language with functional symbols, predicate symbols and the symbol = for the identity. Then, using the notion of signature, we define databases or, in algebraic terms, heterogeneous (operational-relational) algebraic systems. For trees defined on the basis of the many-sorted first-order language we define two kinds of semantics: trees working in identification mode and trees working in retrieval mode. For both of these semantics we introduce the notions soundness, completeness and (partial) equivalence of trees and relating to these notions we formulate theorems of decidability and axiomatizability. The paper concludes with two theorems giving a basis for a (fast) retrieval algorithm using a proof system.

In the following we use the usual set theoretic notions and notations, in particular, if $\mathfrak{S}$ is a collection of sets, then $\bigcup \mathfrak{S}$ denotes the union of all sets $S \in \mathfrak{S}$. The empty set is denoted by ϕ. If S is an arbitrary set, then S^* is the set of all finite sequences of elements of S. As symbol for the empty sequence we use e. A binary relation R over S is an equivalence relation if it is reflexive over S, symmetric and transitive. R is said to be a partial equivalence relation if it is symmetric and transitive only.

2. Signatures and syntax of many-sorted first-order languages

The basis of many-sorted first-order languages and of interpreting structures is the notion of signature.

DEFINITION 2.1.

A quadruple $\Sigma = [S,F,P,TYPE]$ is said to be a signature if and only if S, F, P are pairwise disjoint sets and TYPE is a mapping from $F \cup P$ into S^*, where $TYPE(f) \neq e$ if $f \in F$.

REMARKS.

1. The set S, F, P is called the set of sorts, the set of functional symbols, and the set of predicate symbols, respectively, of the signature Σ. Sorts, functional symbols, and predicate symbols are denoted by s, f, p, respectively, possibly with arbitrary indices.

2. If for $s_1,\ldots,s_n,s \in S$ $(n \geqq 0)$, $f \in F$ and $p \in P$ the equation $TYPE(f) = s_1\ldots s_n s$ and $TYPE(p) = s_1\ldots s_n$, respectively, holds, then $s_1\ldots s_n s$ and $s_1\ldots s_n$ is called the type of f and p, respectively. In the case of the functional symbol f we introduce the input type and the output type of f by the definition $ITYPE(f) =_{def} s_1\ldots s_n$ and $OTYPE(f) =_{def} s$. The output type s of f is also called output sort of f.

3. If for $f \in F$ the input type of f is empty then we call f an individual name (of the sort s if $OTYPE(f) = s$). Σ is said to be operational and relational, respectively, if and only if P is empty and F contains only individual names, respectively.

DEFINITION 2.2.

Given a fixed signature $\Sigma = [S,F,P,TYPE]$. To every sort $s \in S$ we introduce a fixed infinite set V_s of individual variables of the sort s denoted by v^s, x^s, y^s, z^s (possibly with additional indices). We define $V =_{def} \bigcup_{s \in S} V_s$. Starting with the variables of V, with the functional and predicate symbols of Σ and using the logical connections $\sim$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\forall$, $\exists$, the sign = for the identity, the comma , and the brackets ( ) we form terms t (possibly with indices) and formulae A, B, C (possibly with indices) as usual for many-sorted first-order languages.

Because of lack of space we do not explicitely formulate this definition. The many-sorted first-order language defined above is denoted by $\mathcal{L}(\Sigma,V)$.

3. Databases and semantics of many-sorted first-order languages

Firstly, we define the notion of Σ-database (or, in algebraic terms, the notion of Σ-algebraic system).

DEFINITION 3.1.

A quadruple $\mathcal{D} = [\Sigma, \Delta, \Phi, \Pi]$ is said to be a Σ-database (or shortly a database) if and only if the following conditions are satisfied:

1. Σ is a signature, say $\Sigma = [S,F,P,TYPE]$.
2. Δ is a mapping assigning a set Δ_s to every sort $s \in S$.
3. Φ is a mapping assigning a partial function Φ_f from $\Delta_{s_1} \times \ldots \times \Delta_{s_n}$ into Δ_s to every functional symbol $f \in F$ where $TYPE(f) = s_1 \ldots s_n s$ and $n \geq 0$.
4. Π is a mapping assigning a total function Π_p from $\Delta_{s_1} \times \ldots \times \Delta_{s_n}$ into $\{0,1\}$ to every predicate symbol $p \in P$ where $TYPE(p) = s_1 \ldots s_n$ and $n \geq 0$.

REMARKS.

1. The set Δ_s is called the domain of the sort $s \in S$ (with respect to the database $\mathcal{D}$). It can be empty. The elements of Δ_s are called individuals of the sort s and denoted by ξ^s.
2. If P is empty, then $\mathcal{D}$ is called operational. If, additionally, for every functional symbol $f \in F$, the function Φ_f is total, then $\mathcal{D}$ is called total-operational.
3. If F contains only individual names (i.e. nullary functional symbols), then $\mathcal{D}$ is called a relational database.
4. An individual name $\nu \in F$ is said to be $\mathcal{D}$-proper if and only if the (nullary) function Φ_ν is total, i.e. Φ_ν is an element of Δ_s if $OTYPE(\nu) = s$.
5. The case of relations with nullvalues can be built in as follows. For every sort $s \in S$ we adjoin a new element $null^s$ to Δ_s where $null^s \notin \Delta_s$. Then we replace the term

$\Delta_{s_1} \times \ldots \times \Delta_{s_n}$ by $(\Delta_{s_1} \cup \{\text{null}^{s_1}\}) \times \ldots \times (\Delta_{s_n} \cup \{\text{null}^{s_n}\})$ in condition 4 of DEFINITION 3.1. A predicate $\overline{\Pi}_p$ is said to be null-free if and only if for every $\xi^{s_1} \in \Delta_s \cup \{\text{null}^{s_1}\}, \ldots,$ $\xi^{s_n} \in \Delta_{s_n} \cup \{\text{null}^{s_n}\}$, the equation $\overline{\Pi}_p(\xi^{s_1}, \ldots \xi^{s_n}) = 1$ implies $\xi^{s_1} \in \Delta_{s_1}, \ldots, \xi^{s_n} \in \Delta_{s_n}$.

6. The notion of database introduced by DEFINITION 3.1 does not formalize any mechanism of accessibility. But that is even not necessary for the following investigations.

Now, given an arbitrary database $\mathcal{D} = [\Sigma, \Delta, \Phi, \overline{\Pi}]$, where $\Sigma = [S,F,P,\text{TYPE}]$, and a $\mathcal{D}$-evaluation σ, i. e. for every sort $s \in S$, σ is a mapping from the set V_s of all individual variables of the sort s into the set Δ_s of all individuals of the sort s. Let t and A be an arbitrary term and an arbitrary formula of the language $\mathcal{L}(\Sigma,V)$. Then by structural induction on t and A we get the following.

DEFINITION 3.2.

1. $\text{EL}(t,\mathcal{D},\sigma) =_{\text{def}}$ the element of the sort $\text{OTYPE}(t)$ which is assigned to the term t with respect to the interpreting database $\mathcal{D}$ and the $\mathcal{D}$-evaluation σ.

2. $\text{VAL}(A,\mathcal{D},\sigma) =_{\text{def}}$ the logical value of $\{0,1\}$ which is assigned to formula A with respect to the interpreting database $\mathcal{D}$ and the $\mathcal{D}$-evaluation σ.

Because of lack of space we cannot fully formulate these definitions, but we will accentuate the following facts.

REMARKS.

1. If the domain Δ_s is empty, then the evaluation σ is not defined for any individual variable of the sort s.

2. As the mapping σ or the function Φ_f, where $f \in F$, can be partial, the functional EL can be partial, too.

3. For equations $t_1 = t_2$, where t_1 and t_2 are terms of the same sort, we define $\text{VAL}(t_1=t_2,\mathcal{D},\sigma) =_{\text{def}} 1$ if and only if $\text{EL}(t_1,\mathcal{D},\sigma)$ and $\text{EL}(t_2,\mathcal{D},\sigma)$ exist and $\text{EL}(t_1,\mathcal{D},\sigma)=\text{EL}(t_2,\mathcal{D},\sigma)$. Note that this interpretation agrees with the interpretation of so-called "existence-equations" $t_1 \overset{e}{=} t_2$ (see [1,15]). Predicate

formulae $pt_1 \ldots t_n$ are analogously interpreted.

4. If only the individual variable x^s freely occurs in the term t and in the formula A then we say that t and A, respectively, is one-dimensional of the sort s. In this case the element $EL(t, \mathfrak{D}, \sigma)$ and the logical value $VAL(A, \mathfrak{D}, \sigma)$ depends only on $\sigma(x^s)$. Therefore we can denote this element and this value by $EL(t, \mathfrak{D}, \xi^s)$ and $VAL(A, \mathfrak{D}, \xi^s)$, respectively, if $\sigma(x^s) = \xi^s$.

For a given Σ-database $\mathfrak{D}$ and a collection Γ of Σ-database we define as usual:

DEFINITION 3.3.

1. A is a $\mathfrak{D}$-theorem (shortly $\mathfrak{D} \models A$) if and only if for every $\mathfrak{D}$-evaluation σ, $VAL(A, \mathfrak{D}\sigma) = 1$.

2. A is a Γ-theorem (shortly $\Gamma \models A$) if and only if for every database $\mathfrak{D} \in \Gamma$, A is a $\mathfrak{D}$-theorem.

4. The syntax of trees

Let $\Sigma = [S, F, P, TYPE]$ be an arbitrary signature. Consider the many-sorted first-order language $\mathcal{L}(\Sigma, V)$. Now, we introduce the "new" symbol ε, i.e. a symbol not occurring among the symbols of $\mathcal{L}(\Sigma, V)$. It is called the empty name and in contrast to individual names it cannot carry any semantic meaning (with respect to a given Σ-database $\mathfrak{D}$). Extended individual names of the sort s are defined as individual names of the sort s or the empty name ε. For an arbitrary term t the output sort of t is defined as the sort of t if t is an individual variable and as the sort $OTYPE(f)$ if t begins with f.

By the following four generation rules we define trees of the sort $s \in S$. They are denoted by T (possibly with arbitrary indices).

DEFINITION 4.1.

1. If ν is an extended individual name of the sort s, then ν is a tree of the sort s.

2. If A is a one-dimensional formula of the sort s and T_0, T_1 are trees of the sort s, then $A(T_0, T_1)$ is a tree of

the sort s.

3. If T_1, T_2 are trees of the sort s, then (T_1,T_2) is a tree of the sort s.

4. If t is a one-dimensional term of the sort s and the output sort s' and if for $m \geq 1$ the symbols $\nu_1,\ldots,\nu_m$ denote individual names of the sort s' and $T_1,\ldots,T_m$ are trees of the sort s, then $t(\nu_1:T_1,\ldots,\nu_m:T_m)$ is a tree of the sort s.

5. Trees working in identification mode

Now, we define the first interpretation of trees which is called "trees working in identification mode". Given an arbitrary database $\mathfrak{D} = [\Sigma, \Delta, \Phi, \Pi]$, where $\Sigma = [S,F,P,TYPE]$, and a collection Γ of Σ-databases. Furthermore let T be a tree of the sort $s \in S$ and $\xi^s \in \Delta_s$. By structural induction on T we define the set $SEIN(T, \mathfrak{D}, \xi^s)$ of extended individual names of the sort s which is computed by the tree T with respect to the interpreting database $\mathfrak{D}$ and the input element ξ^s.

DEFINITION 5.1.

1. $SEIN(T, \mathfrak{D}, \xi^s) =_{def} \{T\}$ if T is an extended individual name (of the sort s)
2. $SEIN(A(T_o,T_1), \mathfrak{D}, \xi^s)$

$$=_{def} \begin{cases} SEIN(T_o, \mathfrak{D}, \xi^s) & \text{if } VAL(A, \mathfrak{D}, \xi^s) = 0 \\ SEIN(T_1, \mathfrak{D}, \xi^s) & \text{if } VAL(A, \mathfrak{D}, \xi^s) = 1 \end{cases}$$

3. $SEIN((T_1,T_2), \mathfrak{D}, \xi^s) =_{def} SEIN(T_1, \mathfrak{D}, \xi^s) \cup SEIN(T_2, \mathfrak{D}, \xi^2)$
4. $SEIN(t(\nu_1:T_1,\ldots,\nu_m:T_m), \mathfrak{D}, \xi^s)$

$$=_{def} \bigcup \{SEIN(T_\mu, \mathfrak{D}, \xi^s) \mid \mu \in \{1,\ldots,m\},\ \Phi_{\nu_\mu} \text{ is total, } EL(t, \mathfrak{D}, \xi^s) \text{ exists, and } \Phi_{\nu_\mu} = EL(t, \mathfrak{D}, \xi^2)$$

REMARKS.

1. Using the functional SEIN we can interpret a tree working in identification mode as an identification algorithm as follows. Given an "unknown" element $\xi^s \in \Delta_s$, i.e. the name of ξ^s (or any name if several exist) is unknown. To know this element means to know some (or all) $\mathfrak{D}$-proper individual names of ξ^s, i.e. functional symbols $f \in F$ with $TYPE(f) = s$ such

that $\bar{\Phi}_f$ is total and $\bar{\Phi}_f = \xi^s$.

2. In applications we want to use only such trees T that do not compute "wrong" individual names ν, i.e. $\mathcal{D}$-proper individual names with $\bar{\Phi}_\nu \neq \xi^s$ or ν is the symbol ε. This idea leads to two notions of soundness for trees working in identification mode.

3. The claim to know all ($\mathcal{D}$-proper individual) names of an element $\xi^s \in \Delta_s$ leads to two notions of completeness for trees T working in identification mode.

4. Trees of the sort s generated by the rules 1 and 2 of DEFINITION 4.1 are called binary (or logical or deterministic) trees. As the interpretation shows binary trees working in identification mode compute at least one extended individual name, i.e. they cannot catch the phenomenon of synonymy.

5. DEFINITION 5.1.3 shows that the tree construct (T_1, T_2) can be interpreted as an unconditioned branching of the searching algorithm described by (T_1, T_2). This construct gives the possibility for syntactically describing multiple effects.

6. If for $\xi^s \in \Delta_s$ and $\mu \in \{1,\ldots,m\}$ there is no $\mathcal{D}$-proper individual name ν_μ such that $EL(t, \mathcal{D}, \xi^s)$ exists and $\bar{\Phi}_{\nu_\mu} = EL(t, \mathcal{D}, \xi^s)$, then the set $SEIN(t(\nu_1:T_1,\ldots,\nu_m:T_m), \mathcal{D}, \xi^s)$ is empty.

7. If for $\mu_1, \mu_2 \in \{1,\ldots,m\}$ with $\mu_1 \neq \mu_2$ the individual names ν_{μ_1} and ν_{μ_2} are equal, then we can interpret this fact as a syntactical branching of the searching algorithm described by the tree $t(\nu_1:T_1,\ldots,\nu_m:T_m)$. We will speak of semantic branching in this tree (with respect to $\mathcal{D}$ and ξ^s) if for $\mu \in \{1,\ldots,m\}$ there is more than one $\mathcal{D}$-proper individual name ν_μ such that $EL(t, \mathcal{D}, \xi^s)$ exists and $\bar{\Phi}_{\nu_\mu} = EL(t, \mathcal{D}, \xi^s)$.

DEFINITION 5.2.

1. T is said to be sound with respect to $\mathcal{D}$ if and only if for every $\xi^s \in \Delta_s$ and for every $\mathcal{D}$-proper individual name ν (of the sort s), if $\nu \in SEIN(T, \mathcal{D}, \xi^s)$ then $\bar{\Phi}_\nu = \xi^s$.

2. T is said to be strongly sound with respect to $\mathcal{D}$ if and only if T is sound with respect to $\mathcal{D}$ and for every $\xi^s \in \Delta_s$, $SEIN(T, \mathcal{D}, \xi^s)$ does not contain the empty name ε.

3. T is said to be complete with respect to $\mathcal{D}$ if and only if for every $\xi^s \in \Delta_s$ there exists a $\mathcal{D}$-proper individual name ν of the sort s such that $\Phi_\nu = \xi^s$ and $\nu \in SEIN(T, \mathcal{D}, \xi^s)$.

4. T is said to be strongly complete with respect to $\mathcal{D}$ if and only if T is complete with respect to T and for every $\xi^s \in \Delta_s$ and for every $\mathcal{D}$-proper individual name ν of the sort s, if $\Phi_\nu = \xi^s$ then $\nu \in SEIN(T, \mathcal{D}, \xi^s)$.

5. T is said to be sound, ..., strongly complete with respect to Γ if and only if for every database $\mathcal{D} \in \Gamma$, T is sound, ..., strongly complete with respect to $\mathcal{D}$, respectively.

6. If Γ consists of all arbitrary Σ-databases, then we say that T is logically sound, ..., logically strongly complete, respectively, and omit the symbol Γ.

We now define some (partial) equivalence relations for trees working in identification mode. These can be used, for instance, to develop theories of optimization or of inductive inference for trees (working in identification mode).

Given a database $\mathcal{D} = [\Sigma, \Delta, \Phi, \Pi]$ and a collection Γ of databases with the signature Σ. Let T_1 and T_2 be trees of the sort $s \in S$.

DEFINITION 5.3.

1. $T_1 \overset{\sim}{\mathcal{D}} T_2$ if and only if for every $\xi^s \in \Delta_s$ and for every $\mathcal{D}$-proper individual name ν of the sort s, $\nu \in SEIN(T_1, \mathcal{D}, \xi^s)$ if and only if $\nu \in SEIN(T_2, \mathcal{D}, \xi^s)$.

2. $T_1 \overset{\approx}{\mathcal{D}} T_2$ if and only if for every $\xi^s \in \Delta_s$,

$$SEIN(T_1, \mathcal{D}, \xi^s) = SEIN(T_2, \mathcal{D}, \xi^s)$$

3. $T_1 \overset{\approx}{\underset{\mathcal{D}}{\approx}} T_2$ if and only if $T_1 \overset{\approx}{\mathcal{D}} T_2$ and for every $\xi^s \in \Delta_s$, $SEIN(T_1, \mathcal{D}, \xi^s) \cup SEIN(T_2, \mathcal{D}, \xi^s)$ contains only $\mathcal{D}$-proper individual names.

4. $T_1 \overset{\sim}{\Gamma} T_2$, $T_1 \overset{\approx}{\Gamma} T_2$ and $T_1 \overset{\approx}{\underset{\Gamma}{\approx}} T_2$ if and only if for every database $\mathcal{D} \in \Gamma$, $T_1 \overset{\sim}{\mathcal{D}} T_2$, $T_1 \overset{\approx}{\mathcal{D}} T_2$ and $T_1 \overset{\approx}{\underset{\mathcal{D}}{\approx}} T_2$, respectively.

5. If Γ consists of all arbitrary Σ-databases, then we speak of logical equivalence of trees and omit the symbol Γ.

CONCLUSION.

1. $T_1 \overset{\approx}{\underset{\mathfrak{D}}{\approx}} T_2 \Rightarrow T_1 \overset{\approx}{\underset{\mathfrak{D}}{}} T_2 \Rightarrow T_1 \overset{\sim}{\underset{\mathfrak{D}}{}} T_2$

2. $T_1 \overset{\approx}{\underset{\Gamma}{\approx}} T_2 \Rightarrow T_1 \overset{\approx}{\underset{\Gamma}{}} T_2 \Rightarrow T_1 \overset{\sim}{\underset{\Gamma}{}} T_2$

3. $\overset{\sim}{\underset{\mathfrak{D}}{}}$, $\overset{\approx}{\underset{\mathfrak{D}}{}}$, $\overset{\sim}{\underset{\Gamma}{}}$, and $\overset{\approx}{\underset{\Gamma}{}}$ are reflexive, symmetric and transitive, i.e. equivalence relations.

4. $\overset{\approx}{\underset{\mathfrak{D}}{\approx}}$ and $\overset{\approx}{\underset{\Gamma}{\approx}}$ are symmetric and transitive, i.e. so-called partial equivalence relations.

PROOF.

Use definition 5.3.

THEOREM 5.1.

1. If the set of Γ-theorems of the language $\mathcal{L}(\Sigma,V)$ is decidable (recursively enumerable), then the sets of trees characterized by DEFINITION 5.2.5 and the binary relations for trees characterized by DEFINITION 5.3.4 are decidable (recursively enumerable) too, respectively.

2. If DS is a deductive system (possibly with ω-rules) for characterizing the Γ-theorems of $\mathcal{L}(\Sigma,V)$, then for each set of trees characterized by DEFINITION 5.2.5 and for each binary relation for trees characterized by DEFINITION 5.3.4, from DS in a natural way a deductive system (possibly with ω-rules, too) can be derived to characterize this set and this relation, respectively.

PROOF.

1. Use the definitions under consideration.

2. See the full draft version of this paper which will be published later elsewhere.

6. Trees working in retrieval mode

Now, we turn over to define retrieval processes described by trees.

Given a one-dimensional formula A and a tree T, both of the sort $s \in S$. We interpret A as a query or as a request expression of the given database $\mathfrak{D}$, i.e. by definition, we are interested in knowing at least one (or all) proper individual names of every element $\xi^s \in \Delta_s$ with $VAL(A,\mathfrak{D},\xi^s) = 1$.

Therefore we define the set $RET(T, \mathfrak{D}, A)$ of all individual names of the sort s retrieved in the tree T by the query A with respect to the database $\mathfrak{D}$ as follows.

DEFINITION 6.1.

$$RET(T, \mathfrak{D}, A) =_{def} \bigcup \{SEIN(T, \mathfrak{D}, \xi^s) \mid \xi^s \in \Delta_s \text{ and } VAL(A, \mathfrak{D}, \xi^s) = 1\}$$

Let $\mathfrak{R}$ be an arbitrary fixed set of one-dimensional formulae of the sort $s \in S$ from $\mathfrak{L}(\Sigma, V)$. In the following a set of this kind is called request language because only formulae of $\mathfrak{R}$ are admitted as queries. Let x^s be the individual variable (of the sort s) which freely occurs in A.

Analogously to DEFINITION 5.1 we formulate

DEFINITION 6.2.

1. T is said to be retrieval-sound with respect to $\mathfrak{D}$ and $\mathfrak{R}$ if and only if for every $A \in \mathfrak{R}$ and for every $\mathfrak{D}$-proper individual name ν of the sort s, if $\nu \in RET(T, \mathfrak{D}, A)$ then $\mathfrak{D} \models A(^x/_\nu)$.

2. T is said to be strongly retrieval-sound with respect to $\mathfrak{D}$ and $\mathfrak{R}$ if and only if T is retrieval sound and for every $A \in \mathfrak{R}$, the empty name ϵ is not an element of $RET(T, \mathfrak{D}, A)$.

3. T is said to be retrieval-complete with respect to $\mathfrak{D}$ and $\mathfrak{R}$ if and only if for every $A \in \mathfrak{R}$ and $\xi^s \in A_s$, if $VAL(A, \mathfrak{D}, \xi^s) = 1$, then there exists a $\mathfrak{D}$-proper individual name ν of the sort s such that $\Phi_\nu = \xi^s$ and $\nu \in RET(T, \mathfrak{D}, A)$.

4. T is said to be strongly retrieval-complete with respect to $\mathfrak{D}$ and $\mathfrak{R}$ if and only if T is retrieval complete with respect to $\mathfrak{D}$ and $\mathfrak{R}$ and for every $A \in \mathfrak{R}$, for every $\xi^s \in \Delta_s$ and for every $\mathfrak{D}$-proper individual name ν of the sort s, if $VAL(A, \mathfrak{D}, \xi^s) = 1$ and $\Phi_\nu = \xi^s$, then $\nu \in RET(T, \mathfrak{D}, A)$.

5. T is said to be retrieval-sound, ..., strongly retrieval-complete with respect to Γ and $\mathfrak{R}$ if and only if for every database $\mathfrak{D} \in \Gamma$, T is retrieval-sound, ..., strongly retrieval-complete with respect to $\mathfrak{D}$ and $\mathfrak{R}$, respectively.

6. If Γ consists of all arbitrary databases (with the signature Σ), then we say that T is logically retrieval-sound,

..., logically strongly retrieval-complete, respectively, and omit the symbol Γ.

Now, analogously to DEFINITION 5.3, we define the following binary relations for trees working in retrieval mode.

DEFINITION 6.3.

1. $T_1 \widetilde{\mathcal{D},\mathcal{R}} T_2$ if and only if for every $A \in \mathcal{R}$ and for every $\mathcal{D}$-proper individual name of the sort ν, $\nu \in \mathrm{RET}(T_1, \mathcal{D}, A)$ if and only if $\nu \in \mathrm{RET}(T_2, \mathcal{D}, A)$.

2. $T_1 \widetilde{\widetilde{\mathcal{D},\mathcal{R}}} T_2$ if and only if for every $A \in \mathcal{R}$,

$\mathrm{RET}(T_1, \mathcal{D}, A) = \mathrm{RET}(T_2, \mathcal{D}, A)$.

3. $T_1 \widetilde{\widetilde{\mathcal{D},\mathcal{R}}} T_2$ if and only if $T_1 \widetilde{\mathcal{D},\mathcal{R}} T_2$ and for every $A \in \mathcal{R}$, $\mathrm{RET}(T_1, \mathcal{D}, A) \cup \mathrm{RET}(T_2, \mathcal{D}, A)$ contains only $\mathcal{D}$-proper individual names.

4. $T_1 \widetilde{\Gamma,\mathcal{R}} T_2$, $T_1 \widetilde{\widetilde{\Gamma,\mathcal{R}}} T_2$ and $T_1 \widetilde{\widetilde{\Gamma,\mathcal{R}}} T_2$ if and only if for every database $\mathcal{D} \in \Gamma$, $T_1 \widetilde{\mathcal{D},\mathcal{R}} T_2$, $T_1 \widetilde{\mathcal{D},\mathcal{R}} T_2$ and $T_1 \widetilde{\widetilde{\mathcal{D},\mathcal{R}}} T_2$, respectively.

5. If Γ consists of all arbitrary databases with the signature Σ, then we speak of the logical equivalence of trees and omit the symbol Γ.

CONCLUSION

1. $T_1 \widetilde{\widetilde{\mathcal{D},\mathcal{R}}} T_2 \Longrightarrow T_1 \widetilde{\widetilde{\mathcal{D},\mathcal{R}}} T_2 \Longrightarrow T_1 \widetilde{\mathcal{D},\mathcal{R}} T_2$

2. $T_1 \widetilde{\widetilde{\Gamma,\mathcal{R}}} T_2 \Longrightarrow T_1 \widetilde{\widetilde{\Gamma,\mathcal{R}}} T_2 \Longrightarrow T_1 \widetilde{\Gamma,\mathcal{R}} T_2$

3. $\widetilde{\mathcal{D},\mathcal{R}}$, $\widetilde{\widetilde{\mathcal{D},\mathcal{R}}}$, $\widetilde{\Gamma,\mathcal{R}}$ and $\widetilde{\widetilde{\Gamma,\mathcal{R}}}$ are reflexive, symmetric and transitive, i.e. equivalence relations.

4. $\widetilde{\widetilde{\mathcal{D},\mathcal{R}}}$ and $\widetilde{\widetilde{\Gamma,\mathcal{R}}}$ are symmetric and transitive, i.e. so-called partial equivalence relations.

PROOF

Use definition 6.3.

Analogously to THEOREM 5.1 we can prove the following theorem.

THEOREM 6.1.

1. If the set of Γ-theorems of the language $\mathcal{L}(\Sigma, V)$ is decidable (recursively enumerable), then the sets of trees characterized by DEFINITION 6.2.5 and the binary relations for

trees characterized by definition 6.3.4 are decidable (recursively enumerable), too, respectively.

2. If DS is a deductive system (possibly with ω-rules) for characterizing the Γ-theorems of $\mathcal{L}_\Sigma$, then for each set of trees characterized by DEFINITION 6.2.5 and for each binary relation for trees characterized by DEFINITION 6.3.4 from DS in a natural way a deductive system (possibly with ω-rules, too) can be derived to characterize this set and this relation, respectively.

PROOF

1. Use the definitions under consideration.

2. See the full draft version of this paper which will be published later elsewhere.

7. A "logical" retrieval algorithm based on a proof system

By studying the functional properties of the mapping RET we can develop a "logical" retrieval algorithm which is based on a theorem tester or a proof system. The basis of this algorithm is the following theorem formulated with respect to the syntactical definition of trees. Assume that in A, B and t the same individual variable freely occurs.

THEOREM 7.1.

1. $\mathrm{RET}(\nu, \mathfrak{D}, A) = \left\{ \begin{array}{ll} \phi & \text{if } \mathfrak{D} \models \sim A \\ \{\nu\} & \text{if not } \mathfrak{D} \models \sim A \end{array} \right\}$ and ν is an extended individual name.

2. $\mathrm{RET}(B(T_0, T_1), \mathfrak{D}, A) = \mathrm{RET}(T_0, \mathfrak{D}, A \wedge \sim B) \cup \mathrm{RET}(T_1, \mathfrak{D}, A \wedge B)$

3. $\mathrm{RET}((T_1, T_2), \mathfrak{D}, A) = \mathrm{RET}(T_1, \mathfrak{D}, A) \cup \mathrm{RET}(T_2, \mathfrak{D}, A)$

4. $\mathrm{RET}(t(\nu_1 : T_1, \ldots, \nu_m : T_m), \mathfrak{D}, A) = \bigcup_{\mu=1}^{m} \mathrm{RET}(T_\mu, \mathfrak{D}, A \wedge t = \nu_\mu)$

PROOF

Use definition 6.1.

To describe the algorithm we have in mind we need the following two mappings WAY(T) and COND(T,ω), where $\omega \in \mathrm{WAY}(T)$. The first is to be the set of all ways of T, i.e. "connected" sequences of edges of T starting in the root of T and ending in a leaf of T. COND(T,ω) describes the logical condition derived in T along the way $\omega \in \mathrm{WAY}(T)$.

DEFINITION 7.1.

1. $WAY(\nu) =_{def} \{e\}$ if ν is an extended individual name.
2. $WAY(A(T_o,T_1)) =_{def} \{0\omega \mid \omega \in WAY(T_o)\} \cup \{1\omega \mid \omega \in WAY(T_1)\}$
3. $WAY((T_1,T_2)) =_{def} \{1\omega \mid \omega \in WAY(T_1)\} \cup \{2\omega \mid \omega \in WAY(T_2)\}$
4. $WAY(t(\nu_1:T_1,\ldots,\nu_m:T_m)) =_{def} \bigcup_{\mu=1}^{m} \{\mu\omega \mid \omega \in WAY(T_\mu)\}$

DEFINITION 7.2.

1. $COND(\nu,e) =_{def} A_o \rightarrow A_o$ if ν is an extended name, A_o is a fixed arbitrary formula.
2.0. $COND(A(T_o,T_1),0\omega) =_{def} \sim A \wedge COND(T_o,\omega)$
2.1. $COND(A(T_o,T_1),1\omega) =_{def} A \wedge COND(T_1,\omega)$
3.1. $COND((T_1,T_2),1\omega) =_{def} COND(T_1,\omega)$
3.2. $COND((T_1,T_2),2\omega) =_{def} COND(T_2,\omega)$
4. $COND(t(\nu_1:T_1,\ldots,\nu_m:T_m),\mu\omega) =_{def} t=\nu_\mu \wedge COND(T_\mu,\omega)$ if $\mu \in \{1,\ldots,m\}$.

Furthermore, it is clear how the extended individual name $EIN(T,\omega)$ determined as the label of the leaf which is given in the tree T by the way ω has to be defined. Using THEOREM 7.1 and the above definitions we can compute the set $RET(T,\mathfrak{D},A)$ by completely examining the set $WAY(T)$ and checking whether $\mathfrak{D} \models \sim(A \wedge COND(T,\omega))$ or not for $\omega \in WAY(T)$. By THEOREM 7.1 we know that for every extended individual name ν, $\nu \in RET(T,\mathfrak{D},A)$ if and only if there exists a way $\omega \in WAY(T)$ such that $\mathfrak{D} \models \sim(A \wedge COND(T,\omega))$ does not hold, but $EIN(T,\omega) = \nu$.

In many cases the whole set $WAY(T)$ as search space can be restricted by the following theorem.

THEOREM 7.2.

1. $RET(T,\mathfrak{D},A) = \emptyset$ if $\mathfrak{D} \models \sim A$
2. $RET(B(T_o,T_1),\mathfrak{D},A) = RET(T_o,\mathfrak{D},A)$ if $\mathfrak{D} \models A \rightarrow \sim B$
3. $RET(B(T_o,T_1),\mathfrak{D},A) = RET(T_1,\mathfrak{D},A)$ if $\mathfrak{D} \models A \rightarrow B$
4. $RET(t(\nu_1:T_1,\ldots,\nu_m:T_m),\mathfrak{D},A) = \bigcup_{\mu \in M} RET(T_\mu,\mathfrak{D},A)$
   if $\mathfrak{D} \models A \rightarrow \bigvee_{\mu \in M} t=\nu_\mu$ and $M \subseteq \{1,\ldots,m\}$

PROOF

Use the definition of RET.

Finally, we will try to use a given theorem tester or proof

system for checking whether, for instance, $\mathfrak{D} \models \sim(A \wedge COND(T, \omega))$, $\mathfrak{D} \models \sim A$, $\mathfrak{D} \models A \longrightarrow \sim B$, $\mathfrak{D} \models A \longrightarrow B$ or not.

REFERENCES (selected from references of the full draft version)

[1] BURMEISTER, P.: A Model Theoretic Oriented Approach to Partial Algebras. Akademie-Verlag Berlin 1986, 319 pages

[2] CHEN, K. and Z. RAS: DDH-approach to information systems. Proceedings of the 982 CISS in Princeton, N.J., 521-526

[3] CHEN, K. and Z. RAS: Dynamic hierarchical data bases. Proceedings of the 1983 ICAA in Taipei, Taiwan

[4] CHEN, K. and Z. RAS: An axiomatic theory of information trees. The University of North Carolina at Charlotte. Department of Mathematics and Computer Science, 1984

[5] CHEN, K. and Z. RAS: Homogeneous information trees. Fundamenta Informaticae 8,1 (1985), 123-149

[6] JACOBS, B. E.: On database logic. Journal of the ACM 29,3 (April 1982), 310-332

[7] KNUTH, D. E.: The Art of Computer Programming. Vol. 3. Sorting and Searching, 1973

[8] LÜDDE, E.: Zur Optimierung fragebogentheoretisch definierter Suchverfahren. Dissertation A, Humboldt-Universität zu Berlin, 1976

[9] MAREK, W. and Z. PAWLAK: Mathematical foundations of information storage and retrieval I, II, III. CC PAS Reports, I: No. 135, II: No. 136, III: No. 137 (1973), Warsaw

[10] PAWLAK, Z.: Foundations of information retrieval. CC PAS Reports, No. 101, Warsaw 1973

[11] PICARD, C. F.: Theorie des questionnaires, Paris 1965
Graphes et questionnaires. Paris 1972

[12] RAS, Z: An algebraic approach to information retrieval systems, Int. Journal of Comp. and Inf. Sciences, Vol. 11, No. 4, (August 1982), 275-293

[13] SALTON, G.: Automatic Information Organization and Retrieval. New York 1968

[14] SANDEWALL, E.: New Computing Environment, Software System Research Center, Linköping University, Sweden, 1980

[15] THIELE, H.: Wissenschaftstheoretische Untersuchungen in algorithmischen Sprachen I. Theorie der Graphschemata-Kalküle. Berlin 1966

[16] THIELE, H.: On a graph-theoretic realization of retrieval systems. Colloques Internationaux du C.N.R.S., No. 276. THEORIE DE L'INFORMATION, Paris (1977), 417-428

[17] THIELE, H.: On equivalent transformations of binary search trees. "Cinquieme Colloque de Lille Les arbres en algebre et en programmation", Lille (1980), 95-109

Searching and retrieval in databases by trees

H. Thiele

Summary

The paper deals with a logical approach to searching and retrieval in relational databases by trees. The starting point is a many-sorted first-order language with functional symbols, predicate symbols and the symbol = for the identity. Then, using the notion of signature, databases or, in algebraic terms, heterogeneous /operational-relational/ algebraic systems are defined. Two kinds of semantics for trees are defined: trees working in identification mode and trees working in retrieval mode. For both of these semantics the notions of soundness, completenes and /partial/ equivalence are introduced and theorems of decidability and axiomatizability are formulated relating to these notions. The paper concludes with two theorems giving a basis for a /fast/ retrieval algorithm using a proof system.

Keresés és visszakeresés adatbázisokban fák segitségével

H. Thiele

Összefoglaló

A cikk a relációs adatbázisokra vonatkozó keresés és visszakeresés logikai megközelitését tárgyalja. A kiinduló pont egy több-szortu első-rendü nyelv funkcionális szimbólumokkal, predikátum szimbólumokkal és az egyenlőség szimbólummal. Használva a szignatura fogalmát, adatbázist ill. heterogén algebrai rendszereket lehet definiálni. A fákra vonatkozó kétfajta szemantikátdefiniál a szerző: identifikációs módban ill. visszakeresési módban müködő fák. Mindkét szemantikában a következő fogalmakat vezeti be a szerző: megalapozottság, teljesség és /parciális/ ekvivalencia. Ezekre a fogalmakra aztán eldönthetőségi és axiomatizálhatósági tételeket bizonyit. A cikk végén két tétel van, amelyek segitségével gyors visszakeresési algoritmusok adhatók meg.

MTA SZTAKI Közlemények 37/1987 pp. 201-221

THE MEASURE OF COVERING R^n BY LATTICE TRANSLATES OF A SET

B. UHRIN

Computer and Automation Institute
Hungarian Academy of Sciences
1502 Budapest, Pf.63., Hungary

Introduction

Let $b_1, b_2, \ldots, b_n \in R^n$ be linearly independent vectors and Λ be their integer combination (the point lattice). Let the Lebesgue-measure in R^n and the cardinality of a finite set H be denoted by μ and $|H|$, respectively. Denote by A+B the algebraic (Minkowski) sum of the sets $A, B \subset R^n$, in particular $\Delta A = A - A$.

We say that R^n is covered by lattice translates of the set $A \subset R^n$ (shortly R^n is covered by A) if

$$\bigcup_{u \in \Lambda} (A+u) = R^n. \tag{0.1}$$

We say that R^n is almost covered by A if

$$\mu\left(R^n \setminus \bigcup_{u \in \Lambda} (A+u)\right) = 0. \tag{0.2}$$

Denote $P := \{x \in R^n : x = \sum_{i=1}^{n} \lambda_i b_i \ , \ 0 \leq \lambda_i < 1, \ i=1,2,\ldots,n\}$

(a unit cell of Λ, P is usually identified with the quotient space R^n/Λ). Then (0.2) is equivalent to

$$\mu\left(P \setminus \bigcup_{u \in \Lambda} ((A+u) \cap P)\right) = 0 \tag{0.3}$$

By definition, we call the number

$$\alpha(A) := \mu\left(\bigcup_{u \in \Lambda} ((A+u) \cap P)\right) \tag{0.4}$$

"the measure of covering R^n by A".

This work was partially supported by the Hungarian National Science Foundation, Grant No. 1066 /1985

The relation (0.3) shows that R^n is almost covered by A if and only if

(0.5) $\quad \alpha(A) = \mu(P) = d\Lambda = \text{abs.value}(\det(b_1, b_2 \ldots, b_n))$.

Hadwiger [1] called $\alpha(A)$ the "density of covering R^n by A". We think the "measure" is more appropriate here, because the density of covering is usually used for a different notion, see [2], p 183.

It is clear that

(0.6) $\quad \alpha(A) \leq d\Lambda$.

Hadwiger [1] gave a more flexible upper bound for $\alpha(A)$. He proved for any compact body $A \subset R^n$

$$(0.7) \qquad \alpha(A) \leq \frac{\mu(A-A+A) - \mu(A)}{N(A)} \quad ,$$

where N(A) is the number of non-zero lattice points $u \in \Lambda$ such that $A \cap (A+u) \neq \emptyset$ (Hadwiger called this number as "the number of neighbours of A").

In [3] we have sharpened and extended (0.7) proving that for any L-measurable set $A \subset R^n$

$$(0.8) \qquad \alpha(A) \leq \frac{\mu(\partial A \cap \Lambda; A) - \mu(A)}{N(A)} \quad ,$$

where $\mu(\partial A \cap \Lambda; A)$ is a sort of "measure" of A, and this "measure" is $\leq \mu(A-A+A)$ (see [3]).

We gave in [3] also a lower bound for $\alpha(A)$:

$$(0.9) \qquad \alpha(A) \geq \frac{2\mu(A)}{N(A)+2} \quad .$$

The proofs of (0.8) and (0.9) turned to be surprisingly

simple after using "dual" descriptions of quantities $\alpha(A)$ and $N(A)$.

Namely, one can see easily that

(0.10) $$N(A) = |\mathcal{D}A \cap \Lambda| - 1.$$

Further, denote by $\{\beta\}$ the fractional part of the real number β, i.e. the number such that $0 \le \{\beta\} < 1$ and $\beta - \{\beta\}$ is integer. Then define the canonical map $\varphi : R^n \to P$ as

(0.11) $$\varphi(x) := \sum_{i=1}^{n} \{\xi_i\} b_i \quad \text{if} \quad x = \sum_{i=1}^{n} \xi_i \cdot b_i .$$

The canonical projection of $A \subset R^n$ into $P \sim R^n/\Lambda$ is defined as

(0.12) $$\varphi(A) := \bigcup_{a \in A} \varphi(a).$$

One can see easily that

(0.13) $$\varphi(A) = \{ x \in P : (A-x) \cap \Lambda \neq \emptyset \} = \{ x \in P : A \cap (\Lambda + x) \neq \emptyset \}$$

and this show

(0.14) $$\alpha(A) = \mu(\varphi(A)).$$

Using (0.10) and (0.14) the inequality (0.9) boils down to

(0.15) $$|(A-A) \cap \Lambda| \ge 2(\mu(A) / \mu(\varphi(A))) - 1$$

which is an inequality proved earlier in [4] (see also [5], [6]). This inequality is both a sharpening and extension of the Minkowski convex body theorem.

The identity (0.14) makes it possible to use two more known results for lower esetimations of $\alpha(A)$. The results use successive minima of $\mathcal{D}A$ and A, respectively.

(see, [2], [9]). Denote by ν_i the i-th successive minimum of $\mathcal{O}K$, where K is a bounded convex body. Then taking into account (0.12) we have (see inequality (9) in [2], p.54)

$$(0.16) \qquad \mu(\mathcal{O}(\nu_n K)) \geq \mu(K) \cdot \prod_{i=1}^{n} \nu_i .$$

This inequality, which gives the "Minkowski successive minima theorem" at once, goes back to the proof of this theorem given by Esterman [7] and Weyl [8] . Using (0.14) we have a lower estimation for $\alpha(\nu_n K)$. A refinement of (0.16) for symmetric convex bodies K (i.e. such that K=-K) is given in [9] , p.215:

$$(0.17) \qquad \mu(\mathcal{O}(tK)) \begin{cases} = t^n \mu(K) & \text{if } 2t \leq \lambda_1 \\ \geq t^{n-k} \mu(K) \cdot \prod_{i=1}^{k} (\lambda_i/2) & \text{if } \lambda_k \leq 2t \leq \lambda_{k+1} \end{cases}$$

$k=1,2,\ldots,n$

where λ_i is the i-th successive minimum of K, i=1,2,...,n, and $\lambda_{n+1} := +\infty$.

Using again (0.14) we have lower estimations for $\alpha(tK)$ for symmetric convex sets. Similarly to the first case, inequality (0.17) gives the Minkowski successive minima theorem at once.

The aim of this paper is, after introducing so called "projective successive minima", to prove two more inequalities of type (0.17) in terms of projective successiven minima of $\mathcal{O}K$. These minima resemble usual successive minima but does not coincide with them. We get two new type successive minima theorems as colloraries of our results.

1. Some preliminary remarks

First we recall some straightforward identities. The space R^n can be "decomposed" into the sum of Λ and P and using this decomposition one can see easily

$$(1.1) \qquad \mu(A) = \int_P |A \cap (\Lambda+x)|\, dx$$

This identity is true in a much more general setting, see, e.g. [10] p.36.
The identity (0.13) shows that

$$(1.2) \qquad \mu(A) = \int_{\varphi(A)} |(A \cap (\Lambda+x)|\, dx.$$

Denote

$$(1.3) \qquad A_i := \{ x \in P: |A \cap (\Lambda+x)| = i \}, \quad i=0,1,2,\ldots .$$

Then (1.2) is equivalent to

$$(1.4) \qquad \mu(A) = \sum_{i=1}^{\infty} i\, \mu(A_i)$$

and by (0.13) and (0.14) we also have

$$(1.5) \qquad \alpha(A) = \sum_{i=1}^{\infty} \mu(A_i) .$$

The last two identities proved to be useful in investigation of inequalities of type (0.15), non only in R^n but in topological groups as well, [11].
The identities (1.4) and (1.5) imply

$$(1.6) \qquad \alpha(A) = \mu(A) \iff |A \cap (\Lambda+x)| \leq 1 \quad \text{for a.e. } x \in P.$$

On the other hand it is clear that

$$(1.7) \qquad \mathfrak{D} A \cap \Lambda = \{\theta\} \iff |A \cap (\Lambda+x)| \leq 1 \quad \text{for all } x \in P,$$

where θ is the zero of R^n.

We shall need these two implications in the sequel.

Second type identities come from the decompositon of R^n into the direct sum of two linear subspaces, say, $R^n = T \oplus S$. Denote μ_1 and μ_2 the L-measures in T and S, respectively. Then we have

(1.8) $$\mu(A) = \int_T \mu_2(A \cap (S+z))dz$$

and

(1.9) $$\alpha(A) = \int_T \mu_2(\varphi(A) \cap (S+z))dz.$$

Let I be a k-element subset of the set $J := (1,2,\ldots,n)$ and let T:=linear hull $(b_i, i \in I)$, S:=linear hull $(b_i, i \notin I)$, where (b_i) is the defining basis of Λ. Let $L \subset T$, $M \subset S$ be point lattices defined by $(b_i, i \in I)$, $(b_i, i \notin I)$ and $Q \subset P$, $W \subset P$ be their unit cells (See Figure 1).

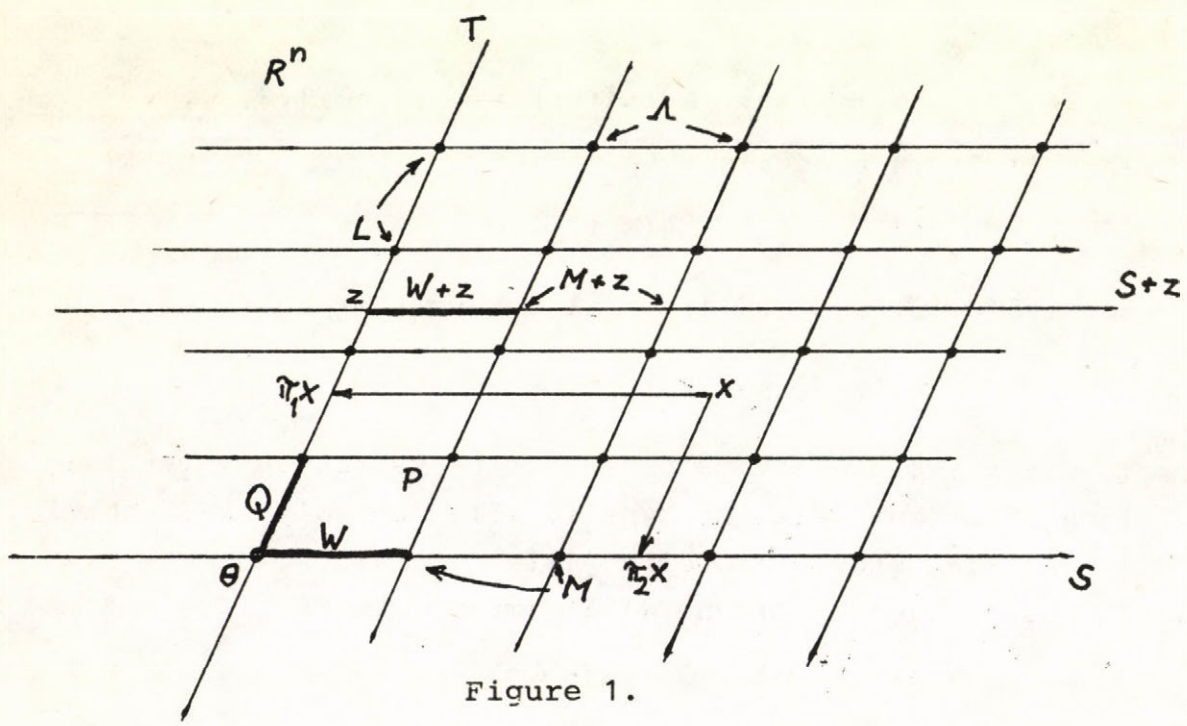


Figure 1.

Let φ_1, φ_2 be the canonical maps $T \to Q$, $S \to W$, and α_1, α_2 be the measures of covering T,S by sets in T, in S. Finally we denote by π_1 and π_2 the projections of $x \in R^n$ into T (along S) into S(along T), respectively, i.e. $x = \pi_1 x + \pi_2 x$, where $\pi_1 x \in T$, $\pi_2 x \in S$ are unique elements (see Figure 1). If we translate S along T, i.e. take S+z, $z \in T$, then the set M+z is a point lattice in S+z when we take the latter mainfold as a space R^{n-k} with z as the origin. The set W+z is a unit cell of M+z. We shall use φ_2 and α_2 for denoting the canonical map S+z $\to$ W+z and the measure of covering S+z by sets in S+z, although strictly speaking these were defined for S and M. This is justified by the fact that if $B \subset S+z$ then the canonical projection of this set into W+z is equal to $\varphi_2(\pi_2(B))+z$ and we shall denote this shortly $\varphi_2(B)$.

One can see easily that $\mu(\varphi(A))$, consenquently $\alpha(A)$, are invariant under translations, i.e.

(1.10) $\mu(\varphi(A)) = \mu(\varphi(A+x))$, $\alpha(A) = \alpha(A+x)$ for all $x \in R^n$.

At the end of this section let us prove

Proposition 0.1. Let $K \subset R^n$ be a convex set containing the origin $\Theta \in R^n$. Then for any $0 \leq \lambda \leq 1$ and any $z \in T$ we have

(1.11) $\alpha_2(K \cap (S+\frac{z}{\lambda})) \geq \alpha_2(\lambda K \cap (S+z))$

where λK is the usual dilatation of K. □

Proof: Let $y_o \in \pi_2(\lambda K \cap (S+z))$ be fixed. We shall prove that

(1.12) $(\frac{1}{\lambda} - 1)\, y_o + \pi_2(\lambda K \cap (S+z)) \subseteq \pi_2(K \cap (S+\frac{z}{\lambda}))$.

To each $y \in \pi_2(\lambda K \cap (S+z))$ there is a unique $x \in K$ such that

$\lambda x=y+z$. Hence $x_o=y_o/\lambda + z/\lambda$ and $x=y/\lambda + z/\lambda$.
The convexity of K implies

$$\lambda x + (1-\lambda)x_o = (\frac{1}{\lambda} - 1)\, y_o + y + \frac{z}{\lambda} \in K,$$

but $y_o, y \in S$, hence

$$(\frac{1}{\lambda} - 1)\, y_o + \pi_2(\lambda K \cap (S+z)) + \frac{z}{\lambda} \subseteq K \cap (S+\frac{z}{\lambda}),$$

and this implies (1.12). Using the monotonity of φ_2, the identity (0.14) (for α_2, φ_2) and the translation invariance of α_2, we get (1.11). ■
One can ask whether instead of (1.9) the identity

$$\alpha(A) = \int_T \alpha_2(A \cap (S+z))\, dz \tag{1.13}$$

holds?
We shall see in the next section that (1.13) is really true for a well defined class of sets.

2. Preparatory lemmas

We use the notations of the previous sections. The proof of our main result depend on the following two lemmas.

Lemma 2.1. If $A \subset R^n$ is a bounded L-measurable set such that

$$\pi_1(\partial A) \cap L = \{\theta\} \quad , \tag{2.1}$$

then

$$\alpha(A) = \int_T \alpha_2(A \cap (S+z))\, dz \quad . \tag{2.2}$$

□

Proof: For better orientation use Figure 1.
By (1.7), the condition (2.1) is equivalent to

(2.3) $\quad |\pi_1(A) \cap (L+y)| \leq 1 \quad$ for all $\quad y \in Q.$

Denote

(2.4) $\quad \psi(z) := \alpha_2(A \cap (S+z)) = \mu_2(\varphi_2(A \cap (S+z))), \quad z \in T.$

According to a well known identity (sse. e.g., [10] p. 36.)

(2.5) $\quad \int_T \psi(z)\,dz = \int_Q (\sum_{v \in L} \psi(v+y))\,dy.$

On the other hand, by (1.9) we have

(2.6) $\quad \alpha(A) = \int_Q \mu_2(\varphi(A) \cap (W+y))\,dy.$

According to (2.3) the basic cell of L can be decomposed

(2.7) $\quad Q = \hat{Q} \cup \tilde{Q},$

where

(2.8) $\quad \hat{Q} := \{ y \in Q : |\pi_1(A) \cap (L+y)| = 1 \},$

(2.9) $\quad \tilde{Q} := \{ y \in Q : \pi_1(A) \cap (L+y) = \emptyset \}.$

The projection $\pi_1(A)$ is defined as

(2.10) $\quad \pi_1(A) := \{ t \in T : A \cap (S+t) \neq \emptyset \}$

and this implies for any $v \in L$

(2.11) $\quad A \cap (S+v+y) \neq \emptyset \iff y \in \hat{Q}.$

Moreover,

(2.12) $\quad \varphi(A) \cap (W+y) \neq \emptyset \iff y \in \hat{Q}.$

Indeed, let $b \in \varphi(A) \cap (W+y)$. Then there are $w \in W$, $a \in A$, $u \in \Lambda$ such that $b = w+y$, $a = u+b$. Hence $a = u+w+y = v+m+w+y$,

where $u=\nu+m, \nu\in L$, $m\in M$, and this shows $\nu+y\in\tilde{\pi}_1(A)\cap(L+y)$. Conversely if $b\in\tilde{\pi}_1(A)\cap(L+y)$, then there are $s\in S$, $a\in A, \nu\in L$, such that $a=s+\nu+y=m+w+\nu+y$, where $s=m+w$, $m\in M$, $w\in W$. Hence $a=u+w+y$, where $u=\nu+m\in\Lambda$, showing that $w+y\in\varphi(A)\cap(W+y)$. The condition (2.8) means that to any $y\in\hat{Q}$ there is only one $\nu\in L$ such that $A\cap(S+\nu+y)\neq\emptyset$. Denote this ν by $\nu(y)$. The above facts imply

$$(2.13)\qquad \int_T \psi(z)\,dz = \int_{\hat{Q}} \psi(\nu(y)+y)\,dy,$$

$$(2.14)\qquad \alpha(A) = \int_{\hat{Q}} \mu_2(\varphi(A)\cap(W+y))\,dy.$$

Now we shall prove that for any $y\in\hat{Q}$

$$(2.15)\qquad \varphi(A)\cap(W+y)=\varphi_2(A\cap(S+\nu(y)+y)-(\nu(y)+y))+y.$$

Let

$$(2.16)\qquad b\in\varphi(A)\cap(W+y).$$

There are $w\in W$, $a\in A$, $\nu\in L$, $m\in M$ such that $b=w+y$, $a=u+b$, $u=\nu+m\in\Lambda$, hence $a=\nu+y+m+w$. So $\nu+y\in\tilde{\pi}_1(A)\cap(L+y)$, implying $\nu=\nu(y)$. One can see easily that

$$(A\cap(S+\nu(y)+y)-(\nu(y)+y))\cap(M+w)\neq\emptyset .$$

This show that

$$(2.17)\qquad w\in\varphi_2(A\cap(S+\nu(y)+y)-(\nu(y)+y)).$$

Let now w be as in (2.17). There are $a\in A$, $s\in S$, $m\in M$, such that $a=s+\nu(y)+y$, $a-(\nu(y)+y)=m+w$. But $a=m+\nu(y)+y+w$, implying that for $b=y+w$ (2.16) holds. This proves (2.15). The identity (2.15) yields

$$(2.18)\qquad \psi(\nu(y)+y)=\mu_2(\varphi(A)\cap(W+y))$$

and taking into account (2.13) and (2.14), we get (2.2). ∎

The above lemma is true under weaker conditions.

Lemma 2.2. If $A \subset R^n$ is a bounded L-measurable set such that

$$\alpha_1(\pi_1(A)) = \mu_1(\pi_1(A)) \tag{2.19}$$

then

$$\alpha(A) = \int_T \alpha_2(A \cap (S+z))dz. \tag{2.20}$$ □

Proof: Denote

$$\underline{Q} := \{y \in Q: |\pi_1(A) \cap (L+y)| \leq 1\}. \tag{2.21}$$

The condition (2.19) is equivalent to (see(1.6))

$$\mu_2(Q \setminus \underline{Q}) = 0. \tag{2.22}$$

Now working with $\underline{Q}$ instead of Q the proof is the same as that of the previous lemma. ■

Lemma 2.3. Let $K \subset R^n$ be a bounded convex set containing the origin $\theta \in R^n$. If

$$\alpha_1(\pi_1(K)) = \mu_1(\pi_1(K)) \tag{2.23}$$

then for $0 \leq \lambda \leq 1$ we have

$$\lambda^k \alpha(K) \geq \alpha(\lambda K). \tag{2.24}$$ □

Proof: Clearly $\pi_1(K)$ is convex and contains the origin $\Theta \in T$, hence denoting $U := \pi_1(K)$,

$$\pi_1(\lambda K) = \lambda \pi_1(K) = \lambda U \subseteq U. \tag{2.25}$$

Denote

$$U_i := \{y \in Q: |U \cap (L+y)| = i\}, \quad i=0,1,2,\ldots. \tag{2.26}$$

The condition (2.23) is equivalent to

(2.27) $\mu_1(U_i) = 0$ for $i=2,3,\ldots$.

The condition (2.25) implies $\lambda U \cap (L+y) \subseteq U \cap (L+y)$ and this implies

(2.28) $(\lambda U)_i \subseteq \bigcup_{j \geq i} U_j$,

consequently

(2.29) $\mu_1((\lambda U)_i) \leq \sum_{j \geq i} \mu_1(U_j)$,

proving that

(2.30) $\mu_1((\lambda U)_i) = 0$ for $i \geq 2$.

This implies, using (2.25) again, that

(2.31) $\alpha_1(\Pi_1(\lambda K)) = \mu_1(\Pi_1(\lambda K))$.

By Lemma 2.2 we have

(2.32) $\alpha(K) = \int_T \alpha_2(K \cap (S+y))\,dy$

and

(2.33) $\alpha(\lambda K) = \int_T \alpha_2(\lambda K \cap (S+z))\,dz$.

Using the inequality (1.11) we can write

(2.34) $\alpha(\lambda K) \leq \int_T \alpha_2(K \cap (S+\frac{z}{\lambda}))\,dz$

and changing the variable in the last integral, i.e. taking $y=z/\lambda$, $dz=\lambda^k\,dy$, we get

(2.35) $\alpha(\lambda K) \leq \lambda^k \int_T \alpha_2(K \cap (S+y))\,dy$. ■

Using the same proof and Lemma 2.1 we get. (Of course, the condition (2.36) is stronger than (2.23).)

Lemma 2.4. Let $K \subset R^n$ be a bounded convex set containing the origin $\theta \in R^n$. If

(2.36) $\Pi_1(\partial K) \cap L = \{\theta\}$

then for $0 \leq \lambda \leq 1$ we have

(2.37) $\qquad \lambda^k \alpha(K) \geq \alpha(\lambda K)$. □

3. The main results

The notations are as in previous sections. First we define two sequences of numbers associated to a set $A \subset R^n$.

Definition 2.1. Let $t_0(A) := +\infty$ and for $1 \leq k \leq n$

$t_k(A) := \sup\{t : t \geq 0$, there is $I \subseteq J$, $|I| = k$ such that

$\alpha_1(\pi_1(tA)) = \mu_1(\pi_1(tA))\}$. □

Definition 2.2. Let $s_0(A) := +\infty$ and for $1 \leq k \leq n$ $s_k(A) := \sup\{t : t \geq 0$, there is $I \subseteq J$, $|I| = k$ such that $\pi_1(\partial tA) \cap L = \{\theta\}\}$. □

These might be called "projective successive minima of A" (for a justification, see, (3.6), (3.8) below).
Using (1.6) and (1.7) we see that

(3.1) $\qquad s_k(A) \leq t_k(A)$.

Both sequences are non-increasing in k, i.e.

(3.2) $\qquad 0 \leq t_n(A) \leq t_{n-1}(A) \leq \ldots \leq t_1(A) \leq t_0(A) = +\infty$,

(3.3) $\qquad 0 \leq s_n(A) \leq s_{n-1}(A) \leq \ldots \leq s_1(A) \leq s_0(A) = +\infty$.

Let us write usual successive minima of ∂A:

(3.4) $\qquad \nu_k(A) := \inf\{t : t \geq 0, \dim(\partial tA \cap \Lambda) \geq k\}$,

where dim (H) is the linear dimension of H, i.e. the

maximal number of linear independent vectors contained in H. Let us write some equivalent forms of s_k and t_k (we use (1.6), (1.7)).

(3.5) $s_k(A) = \sup\{ t: \exists\ I \subseteq J,\ |I| = k,\ \text{s.t.}$

$$|\Pi_1(tA) \cap (L+y)| \leq 1 \quad \forall\, y \in Q \},$$

(3.6) $s_k(A) = \inf\{ t: |\Pi_1(\oplus tA) \cap L| \geq 2 \text{ for all}$

$$I \subseteq J,\ |I| = k\},$$

(3.7) $t_k(A) = \sup\{ t: \exists\ I \subseteq J,\ |I| = k \text{ s.t.}$

$$|\Pi_1(tA) \cap (L+y)| \leq 1 \quad \text{for a.e. } y \in Q \},$$

(3.8) $t_k(A) = \inf\{ t: \text{for any } I \subseteq J,\ |I| = k \text{ there is } H \subseteq Q,$

$$\mu_1(H) > 0 \text{ such that}$$

$$|\Pi_1(tA) \cap (L+y)| \geq 2 \quad \text{for } y \in H \}.$$

One can see easily that

(3.9) $$s_{n-k}(A) \leq \nu_{k+1}(A).$$

The main results of the paper are the following two theorems

<u>Theorem 3.1.</u> Let $K \subset R^n$ be a bounded convex set. Then

(3.10) $$\alpha(tK) \begin{cases} = t^n \mu(K) & \text{if} \quad 0 \leq t \leq s_n(K) \\ \geq t^k \mu(K) \prod_{i=k+1}^{n} s_i(K) & \text{if } s_{k+1}(K) \leq t \leq s_k(K) \end{cases}$$

$$k = 0, 1, \ldots, n-1. \qquad \square$$

Theorem 3.2. Let $K \subset R^n$ be a bounded convex set. Then

$$(3.11)\qquad \alpha(tK) \begin{cases} = t^n \mu(K) & \text{if } 0 \le t \le t_n(K) \\ \ge t^k \mu(K) \prod_{i=k+1}^{n} t_i(K) & \text{if } t_{k+1}(K) \le t \le t_k(K) \end{cases}$$

$k=0,1,\ldots,n-1.$ □

Proof of Theorem 3.1: If $\mu(tK)=0$ then also $\alpha(tK)=0$ and (3.10) holds trivially. Let $\mu(K)>0$. All quantities involved are translation invariant, hence we can assume that $\Theta \in K$. Further, one can easily see that all quantitites are invariant under the closure operation, i.e. $\alpha(tK)=\alpha(t\bar{K})$, $\mu(K)=\mu(\bar{K})$ and $s_k(K)=s_k(\bar{K})$, where ($\bar{K}$ is the) topological closure of K. This implies that also $\alpha(tK_o)=\alpha(tK)$, $\mu(K_o)=\mu(K)$ and $s_k(K_o)=s_k(K)$, where $K_o \subseteq K$ is an open convex set such that $\bar{K}_o=\bar{K}$ (the open kernel of K). Hence one can assume tahat K is open. Denote for short $s_k=s_k(K)$. One can see easily that for open K the supremum in the definition of s_k is attained, i.e. there is $I \subseteq J$, $|I|=k$, such that

$$(3.12)\qquad \pi_I(\mathcal{O}(s_k K)) \cap L = \{\Theta\}.$$

Indeed, the set $\pi_I(\mathcal{O}(s_k K))$ is open, hence any lattice point in it is an inner point, consenquently if

$$(3.13)\qquad |\pi_I(\mathcal{O}(s_k K)) \cap L| > 1$$

for all $I \subseteq J$, $|I| = k$, then for a sufficiently small $\varepsilon > 0$ also

$$(3.14)\qquad |\pi_I(\mathcal{O}((s_k-\varepsilon)K)) \cap L| > 1$$

for all $I \subseteq J$, $|I| = k$, a contradiction.
For all $0 \leq t \leq s_n$ we have $t\mathfrak{B}K \cap \Lambda = \{\theta\}$. This implies using (1.6), (1.7) that $\alpha(tK) = \mu(tK)$, hence (3.10) is true. Assume, that we have proved (3.10) for $t=s_{k+1}$. If $s_{k+1}=s_k$ then $\alpha(s_kK) = \alpha(s_{k+1}K) \geq s_k^{k-1}\mu(K)\prod_{i=k}^{n} s_i$, so (3.10) is true for $t=s_k$ as well. Let $s_{k+1} < s_k$ and $t=\tau s_{k+1}$, $1<\tau\leq s_k/s_{k+1}$. Hence $\exists\, I\subseteq J$, $|I|=k$ such that $\pi_I(\mathfrak{B}tK)\cap L=\{\theta\}$. By Lemma 2.4 we have

$$(3.15)\qquad \left(\frac{1}{\tau}\right)^k \alpha(tK) \geq \alpha\left(\frac{1}{\tau}tK\right) = \alpha(s_{k+1}K) \geq$$

$$\geq s_{k+1}^k \mu(K) \prod_{i=k+1}^{n} s_i = \left(\frac{t}{\tau}\right)^k \mu(K) \prod_{i=k+1}^{n} s_i,$$

that proves (3.10). ■
The proof of Theorem 3.2 is analogous, now we have to use Lemma 2.3.

4. Concluding remarks

Inequalities (3.10) and (3.11) yield at once "theorems on projective successive minima", analogous to the classical Minkowski theorem.
Namely, applying (3.10) and (3.11) to $k=1$, $t=s_1(K)$, $t=t_1(K)$ we get

$$(4.1)\qquad d\Lambda \geq \mu(K)\prod_{i=1}^{n} s_i(K),$$

$$(4.2)\qquad d\Lambda \geq \mu(K)\prod_{i=1}^{n} t_i(K).$$

For $t=1$ the inequalities yield

$$(4.3)\qquad \alpha(K)\begin{cases} = \mu(K) & \text{if } 1 \le s_n(K) \\ \ge \mu(K)\prod_{i=k+1}^{n} s_i(K), & \text{if } s_{k+1}(K) \le 1 \le s_k(K) \end{cases}$$

$$k=0,1,\dots,n-1.$$

$$(4.4)\qquad \alpha(K)\begin{cases} = \mu(K) & \text{if } 1 \le t_n(K) \\ \ge \mu(K)\prod_{i=k+1}^{n} t_i(K), & \text{if } t_{k+1}(K) \le 1 \le t_k(K) \end{cases}$$

$$k=0,1,\dots,n-1.$$

The proof of (0.17) which can be found in [9] pp. 215-218, seems to work also in non-symmetric case, i.e. we think the following sharpening of (0.16) is true: for any bounded convex set $K \subset R^n$ we have (here $\nu_{n+1}(K) := \infty$),

$$(4.5)\qquad \alpha(tK)\begin{cases} = t^n \mu(K) & \text{if } 0 \le t \le \nu_1(K) \\ \ge t^{n-k}\mu(K)\prod_{i=1}^{k} \nu_i(K), & \text{if } \nu_k(K) \le t \le \nu_{k+1}(K) \end{cases}$$

$$k=1,\dots,n.$$

Similarly to (4.3) we can write

$$(4.6)\qquad \alpha(K)\begin{cases} = \mu(K) & \text{if } 1 \le \nu_1(K) \\ \ge \mu(K)\prod_{i=1}^{k} \nu_i(K) & \text{if } \nu_k(K) \le 1 \le \nu_{k+1}(K) \end{cases}$$

$$k=1,\dots,n.$$

We have seen that $s_k(K) \leq t_k(K)$ and $s_{n-k}(K) \leq \nu_{k+1}(K)$, so principally we would compare the results (4.3), (4.4) and (4.6) (in the sense, which is sharper than the other). The comparison is not easy, because it depends not only on successive minima in themselves, but also on their "intervals", say, how $[s_{n-k+1}, s_{n-k}]$ can be compared with $[\nu_k, \nu_{k+1}]$.

Although we were speaking all through the paper about "the" defining basis $B=(b_1, b_2, \ldots, b_n)$ of Λ, it is clear that B can be any basis of Λ. The numbers s_k and t_k defined above depend on B, consenquently the lower bounds in (3.10), (3.11) also depend on B. One can see easily that the proofs of (3.10) and (3.11) work also using a following modified definition of s_k and t_k.

$s_k(A) := \sup \{ t:$ there is a basis B of Λ and $I \subseteq J$, $|I| = k$ s.t. $\pi_I(\partial tA) \cap L = \{\theta\} \}$

$t_k(A) := \sup \{ t:$ there is a basis B of Λ and $I \subseteq J$, $|I| = k$ s.t. $\alpha_I(\pi_I(tA)) = \mu_I(\pi_I(tA)) \}$

These numbers are independent of a particular basis already, they depend on A and Λ only.

The relations (3.1), (3.2) and (3.3) clearly hold again. The relation (3.9) holds as well, but for the proof of it one has to use a well known result in geometry of numbers about the "extendeability" of given linearly independent vectors of Λ to a basis of Λ (after a regural transformation, see [9], p.13, Corollary 2.)

References

[1] H. Hadwiger, Überdeckung des Raumes durch translationsgleiche Punktmengen und Nachbarnzahl, Monatsh.f.Math., 73(1969), 213-217.

[2] C.G. Lekkerkerker, "Geometry of Numbers", Biblio. Math., Vol VIII, North-Holland, Amsterdam-New York, 1969.

[3] B. Uhrin, A remark to the paper of H. Hadwiger "Überdeckung des Raumes durch translationsgleiche Punktmenge und Nachbarnzahl", submitted to Monatsh. f.Math.

[4] B. Uhrin, Some useful estimations in geometry of numbers, Periodica Math. Hungar., 11(2)(1980),95-103.

[5] B. Uhrin, On a generalization of Minkowski convex body theorem, J. Number Th., 13(2)(1981),192-209.

[6] B. Uhrin, Minkowski convex body theorem and the measure of covering R^n by a set, Proc. of Comput. and Automation Inst. of the Hung. Acad.Sci., 29/1983, 115-121.

[7] T. Esterman, Note on a theorem of Minkowski, J. London Math. Soc., 21(1946), 179-182.

[8] H. Weyl, On geometry of numbers, Proc. London Math. Soc., (2),47(1942),21-39.

[9] J.W.S.Cassels, "An Introduction to the Geometry of Numbers", Grundl.Math.Wiss.,99, Springer, Berlin-New York, 1959.

[10] A. Weil, "Basic Number Theory", Grundl. Math. Wiss., 144, Springer, Berlin-New York, 1967.

[11] B. Uhrin, Some remarks about the lattice points in difference sets, in: J. Szabados,Ed.: Proc. of the A. Haar Memorial Conference, Budapest, 1985, Coll.Math.Soc.J.Bolyai, Vol 49, North-Holland, Amsterdam-New York, 1987, 929-937.

The measure of covering R^n by lattice translates of a set

B. Uhrin

Summary

Given a discrete subgroup $\Lambda \subset R^n$ of full dimension (point lattice), its unit cell P and a set $A \subset R^n$, the volume (L-measure) V of the set $\bigcup_{u \in \Lambda} ((A+u) \cap P)$, denoted by $\alpha(A)$, is called the measure of covering R^n by (lattice translates of) A. In the paper two sufficient conditions are given for $\alpha(\lambda K) \leq \lambda^k \alpha(K)$ to hold for all $0 \leq \lambda \leq 1$, where K is a convex set and $0 < k < n$. Using these conditions, so called projection successive minima of A are introduced and it is proved that $\alpha(tK) \geq t^k \cdot V(K) \cdot \prod_{i=k+1}^{n} t_i$ and $\alpha(tK) \geq t^k \cdot V(K) \cdot \prod_{i=k+1}^{n} s_i$ if $t_{k+1} \leq t \leq t_k$ and $s_{k+1} \leq t \leq s_k$, respectively. Here t_i, s_i are the mentioned projection successive minima of K. The results add to a classical result proved for symmetric convex sets and their usual successive minima.

Az R^n lefedésének mértéke egy halmaz rácsszerü eltoltjaival

B. Uhrin

Összefoglaló

Adva van a $\Lambda \subset R^n$ teljes dimenziós diszkrét részcsoport /pont-rács/, P egységcellája és az $A \subset R^n$ halmaz. Az $\bigcup_{u \in \Lambda} ((A+u) \cap P)$ halmaz V térfogatát /L-mértékét/, amelyet $\alpha(A)$-val jelölünk, az R^n A általi lefedése mértékének nevezzük. A cikkben két elégséges feltételét adjuk meg annak, hogy $\alpha(\lambda K) \leq \lambda^k \alpha(K)$ teljesüljön minden $0 \leq \lambda \leq 1$-ra, ahol K konvex halmaz. Használva ezeket az elégséges feltételeket, kétfajta u.n. projekciós szikcesz-sziv minimumot vezetünk be és bebizonyitjuk, hogy $\alpha(tK) \geq t^k V(K) \prod_{i=k+1}^{n} t_i$ ill. $\alpha(tK) \geq t^k V(K) \prod_{i=k+1}^{n} s_i$ ha $t_{k+1} \leq t \leq t_k$ ill. $s_{k+1} \leq t \leq s_k$. Itt t_i és s_i a K projekciós szukcessziv minimumai. Az eredmény adalék egy klasszikus eredményhez, amelyet szimmetrikus konvex halmazokra és a szokásos szukcessziv minimumaira bizonyitottak be.

MTA SZTAKI Közlemények 37/1987 pp. 223-248

RETRIEVAL FROM FUZZY DATABASE BY FUZZY RELATIONAL ALGEBRA

LE TIEN VUONG

Institute of Informatics and Cybernetics,
Hanoi, Viet Nam

HO THUAN

Computer and Automation Institute,
Hungarian Academy of Sciences

1. INTRODUCTION

In the real world, there exists many things and events which can not or need not be precisely defined. This imprecision may be classified into two types: randomness and fuzziness. The concept of randomness is associated with objective things and has been studied by probability theory. The fuzziness is related to subjective phenomena, and has been studied by fuzzy sets theory. The concept of fuzzy sets has been introduced by L. Zadeh/75/, and applied to various fields.

By fuzzy data, we mean data which have the properties of fuzziness, that is, related to human beings or a result of their subjective observations.

In order to represent and to process fuzzy data in a digital computer we must first choose a suitable mathematical model. There are some models for representation of fuzzy data, for example, fuzzy sets and linguistic variable /L. Zadeh 75/, fuzzy graphs/M. Umano 79/ and fuzzy data classification /W. Belke, et al. 78/. From the view-point of easy implemen-

This work was partially supported by the Hungarian National Science Foundation, Grant No. 1066 / 1985

tation and wide application for most database systems, we extend the Cood's model to a relational fuzzy database. The concept of a relational fuzzy database and a somewhat different point of view about the normalization in this model will be discussed detaily in /LE and HO/. In this paper we only introduce a concept of extended relational model with corresponding relational operations for fuzzy relations. These relational operations have been written in PL/1 and experimentalized on an EC1022 computer.

The physical and logical representation of fuzzy data of this system can be seen in /Le Thien Vuong 83,85/.

2. THE CONCEPT OF FUZZY SETS

In this section we shall present briefly the concept of fuzzy sets which are required in the sequel. Here we only want to repeat some definitions of fuzzy sets theory. More details of the discussion may be seen in /L.A. Zadeh, 75/.

We introduce the following definitions.

Definition 2.1

Let U = {u} be a universe of discourse /i.e., a collection of objects/, denoted generically by u; then a fuzzy set $\underset{\sim}{A}$ of U is a set of ordered pairs $\{(\mu_{\underset{\sim}{A}}(u)/u)\}$, $u \in U$, where $\mu_{\underset{\sim}{A}}(u)$ is the grade of membership of u in $\underset{\sim}{A}$ and $\mu_{\underset{\sim}{A}}: U \to [0,1]$ is the membership function.

In order to capture more information about the entities from a mini-world we choose fuzzy sets to apply to our data base.

Let us consider a complete and distributive lattice V which is the set of all fuzzy sets of the universe of discourse U

$$V(U) = \{\mu / \mu : U \to [0,1] \}$$

/see Negoita et al. 75, pp. 15-16/.

In this lattice are defined the operations union, intersection and complementation as follows:

Definition 2.2

Let $\underset{\sim}{A}$ and $\underset{\sim}{B}$ be two fuzzy sets of U.

a/ UNION /denoted by $\underset{\sim}{A} \cup \underset{\sim}{B}$ or by $\underset{\sim}{A}\underset{\sim}{B}$/

The union of two fuzzy sets $\underset{\sim}{A}$ and $\underset{\sim}{B}$ of U is defined

$$\mu_{\underset{\sim}{A}\cup\underset{\sim}{B}}(u) = \text{Max}(\mu_{\underset{\sim}{A}}(u), \mu_{\underset{\sim}{B}}(u)) \quad \text{for} \quad u \in U$$

or

$$\mu_{\underset{\sim}{A}\cup\underset{\sim}{B}}(u) = \mu_{\underset{\sim}{A}}(u) \vee \mu_{\underset{\sim}{B}}(u) \quad \text{for} \quad u \in U;$$

b/ INTERSECTION /denoted by $\underset{\sim}{A} \cap \underset{\sim}{B}$/

The intersection of two fuzzy sets $\underset{\sim}{A}$ and $\underset{\sim}{B}$ of U is defined by

$$\mu_{\underset{\sim}{A}\cap\underset{\sim}{B}}(u) = \text{Min}(\mu_{\underset{\sim}{A}}(u), \mu_{\underset{\sim}{B}}(u)) \quad \text{for} \quad u \in U$$

or

$$\mu_{\underset{\sim}{A}\cap\underset{\sim}{B}}(u) = \mu_{\underset{\sim}{A}}(u) \wedge \mu_{\underset{\sim}{B}}(u) \quad \text{for} \quad u \in U$$

c/ COMPLEMENTATION

$\neg\underset{\sim}{A}$ is the complementation of fuzzy set $\underset{\sim}{A}$ of U defined by

$$\mu_{\neg \underset{\sim}{A}}(u) = 1 - \mu_{\underset{\sim}{A}}(u) \qquad \text{for} \quad u \in U;$$

d/ EQUALITY

$$\underset{\sim}{A} = \underset{\sim}{B} \Leftrightarrow \mu_{\underset{\sim}{A}}(u) = \mu_{\underset{\sim}{B}}(u) \qquad \text{for} \quad u \in U;$$

e/ CONTAINMENT

$$\underset{\sim}{A} \subseteq \underset{\sim}{B} \Leftrightarrow \mu_{\underset{\sim}{A}}(u) \leq \mu_{\underset{\sim}{B}}(u) \qquad \text{for} \quad u \in U.$$

The symbols $\wedge$, $\vee$ and $\neg$ stand for minimum, maximum and complement.

Remark

The symbols $\wedge$, $\vee$ and $\neg$ stand for AND, OR and NOT in ordinary logic.

Definition 2.3

Let $U = \{u\}$ and $V = \{v\}$ be two arbitrary universes of discourse. A fuzzy /binary/ relation R from U to V is a fuzzy subset of the Cartesian product $U \times V = \{(u,v)\}$, characterized by the membership function $\mu_R : U \times V \to [0,1]$.
A fuzzy relation R in $U \times V$ is expressed by

$$R = \{(\mu(u_1,v_1) / (u_1,v_1), \ldots, \mu(u_n,v_m)/(u_n,v_m) : u_i \in U, v_j \in V\}$$

Remark

Generally, an n-ary fuzzy relation R in $U_1 \times \ldots \times U_n$ is characterized by a membership function over $U_1 \times \ldots \times U_n$ /see L.A. Zadeh 75/.

Definition 2.4

A linguistic variable is characterized by a quintuple (A,T,U,G,M) in which A is the name of the variable; T denotes the term set of A, that is, the set of names of linguistic values of A; G is the syntactic rule for generating the names in T and M is the semantic rule for associating with each t in T its meaning M(t), which is a fuzzy set of U.

The meaning of a linguistic value t can be defined as

$$M(t) = \{(\mu_t(u) \,/u) : u \in U\} .$$

3. RELATIONAL MODEL AND EXTENDED RELATION

3.1 The data model

The relational operations developed in this paper are applicable to most database systems, specially to fuzzy database systems. The concept of relational model may be seen in /E.F. Cood, 70/.

To simplify later discussions, we shall choose the following sample data model of a system consisting of two n-ary normalized relations /fig. 1 and fig. 2/.

Figure 1 represents some entries of the relation

EMP(MNO,MNAME,AGE,DNO,SAL),

where every row of the table corresponds to the data of an object /an employee/. The elements of EMP consists of an employee's person ordinal number, name, age, department ordinal number and his salary. Figure 2 represents some entries of the relation

DEPT(DNO,DNAME)

where a row of the DEPT-relation consists of the department ordinal number and the name of a department.

EMP (MNO	NAME	AGE	DNO	SAL)
100	Fischer	25	10	1000
101	Neuman	20	11	1500
102	King	20	11	2000
103	Shmid	30	12	2000
104	John	50	13	3000

Fig.1: EMP-relation

DEPT (DNO	DNAME)
10	A
11	B
12	C
13	D

Fig.2: DEPT-relation

Such a systems should be limited to consist of only precise data. There exists yet another possibility for describing a relation in which some linguistic values /fuzzy term/ have been used as fuzzy data /see Fig. 3/.

F-EMP (MNO	NAME	AGE	DNO	SAL)
100	Fischer	25	10	1000
101	Neuman	young	11	1500
102	King	young	11	high
103	Shmid	30	12	2000
104	John	OLD	13	very high

Fig.3: Extended relation F-EMP

In /V. Tahani, 77/ the author had given a method for fuzzy queries processing based on a data model with precise situations. In /M. Umano, 79/ the author had introduced a logical and physical representation for the fuzzy sets as the fuzzy data. These fuzzy sets have been processed by 52 operations.

In this paper we consider not only precise /certain/ data but also fuzzy data. This data can be processed by relational operations.

3.2 Extended relational model

Following customary notation, we use the letters A,B,... to denote single attributes, X,Y,... to denote sets of attributes. ER to denote an extended relation and ATTR(ER) to denote the attribute set of the relation ER. Let $U(A_i)$ or U_i be the usual regular domain of an attribute A_i. The set U_i must be added by the corresponding set of fuzzy term, i.e. the linguistic values of A_i /see Def. 2.4/ in order to enable to process the data represented in Fig. 3.

Every $t \in T$ can be evaluated by semantic rules /see Def. 2.4/ into a fuzzy set. Then the extended domain of attribute A_i is $D(A_i) = U(A_i) \cup T(A_i)$ or briefly $D_i = U_i \cup T_i$.
In order to describe easily the nonprocedural sublanguage as SEQUEL, we assume that the data of the system have the logical and physical representation as in /Le Tien Vuong, 83,85/.

We can formulate an extended relational model as follows.

Definition 3.1

An extended database D_e is defined as a collection of extended relations

$$D_e = \{ER_1,\ldots,ER_n\},$$

where every extended relation is a subset of Cartesian product of the extended domains

$$ER_i \subseteq \{U(A_{i_1}) \cup T(A_{i_1})\} x \ldots x \{U(A_{i_{m_i}}) \cup T(A_{i_{m_i}})\}$$

in which $U(A_{i_j})$ or U_{i_j}, $j = 1,\ldots,m_i$ are basic domains and $T(A_{i_j})$ or T_{i_j} are sets of fuzzy terms of the corresponding attributes A_{i_j}.

To illustrate the Definition 3.1 we consider the following example /see Fig. 3/. The extended relation F-EMP consists of the following domains:

U_1(MNO) : all 6-space digits,

U_2(NAME) : all possible individual names,

U_3(AGE) : all integers 1-150,

T_3(AGE) : the set of all fuzzy terms as ols, young,...

U_4(DNO) : all possible 2-space digits,

U_5(SAL) : all 4-space digits,

T_5(SAL) : the set of all fuzzy term as high, middle,...

We have

$$F\text{-}EMP \subseteq U_1 \times U_2 \times \{U_3 \cup T_3\} \times U_4 \times \{U_5 \cup T_5\}.$$

Here the sets T_1,T_2 and T_4 are empty.

3.3 The relational operations on an extended relation

The purpose of this section is to extend the operations of relational algebra to extended relations in order to process a class of fuzzy queries. These fuzzy queries should be processed not only by operations of fuzzy sets but also by relational operations.

Let r denote a tuple of ER; $A_i \in ATTR(ER)$; $r[A_i]$ is the value of attribute A_i in a tuple $r \in ER$ with $r[A_i] = a_{ij} \in D(A_i)$. If $X = \{A_1,\ldots,A_k\}$ is a subset of ATTR(ER) and r is a tuple over ATTR(ER) then the restriction of r on X will be called the X-value of r denoted by $r[X]$ and is expressed by

$$r[X] = r[A_1,\ldots,A_k] = (r[A_1],\ldots,r[A_k]).$$

Based on Definition 2.2 and Definition 2.4, we can introduce the following concepts.

If $r_1[A] = u_1$, $r_2[A] = u_2$, $r_1, r_2 \in ER$ then the equality of $r_1[A]$ and $r_2[A]$ is defined by

$$r_1[A] = r_2[A] \Leftrightarrow u_1 = u_2 \quad \text{if} \quad u_1, u_2 \in U(A) \quad \text{or}$$

$$M(u_1) = M(u_2) \quad \text{if} \quad u_1, u_2 \in T(A).$$

In general case, where $X \subseteq ATTR(ER)$ the equality of $r_1[X]$ and $r_2[X]$ can be defined analogously:

Let be $r_1[X] = (r_1[A_1], \ldots, r_1[A_k])$,

$r_2[X] = (r_2[A_1], \ldots, r_2[A_k])$,

$r_i[A_j] \in D_j$, $i = 1,2$; $j = 1, \ldots, k$.

The equality of $r_1[X]$ and $r_2[X]$ is defined by

$$r_1[X] = r_2[X] \Leftrightarrow r_1[A_i] = r_2[A_i] \text{ for } i = 1, \ldots, k;$$

$$A_i \in ATTR(ER).$$

In the following, we introduce some definitions of extended relational operations.

a. Projection

Based on the above concepts the projection of an extended relation should be defined as follows:

Definition 3.2

Let ER be an extended relation over a set of attributes $ATTR(ER)$ and $X \subseteq ATTR(ER)$ with $X = \{A_1, \ldots, A_k\}$.
The projection of ER over X is defined by

$$ER[X] = \{r[X]: r \in ER\}$$

$$= \{(r[A_1], \ldots, r[A_k]): r \in ER \wedge A_i \in X \wedge X \subseteq ATTR(ER) \wedge i=1,\ldots,k.\}$$

In more intuitive terms, ER[X] is obtained from ER by deleting all columns of ER not corresponding to attributes in X and identifying duplicate tuples in what remains.

Remark

The Projection introduced here is analogous to Codd's projection /see E.F. Codd, 70/. The difference consists only in that two tuples are equal if all corresponding components of tuples are equal in the sense of fuzzy sets /see Definition 2.2/.

We have now a simple example /see Figure 3/.

Example

Let X = {AGE, DNO}

F-EMP[X] =

(AGE	DNO)
25	10
young	11
30	12
old	13

b. Join

The equijoin of two extended relations is defined as follows:

Definition 3.3

Let ER and ES be two extended relations over the attributes sets ATTR(ER) and ATTR(ES), respectively. A∈ATTR(ER) and B∈ATTR(ES) are two attributes. The equijoin of ER and ES on A and B is defined by

$ER[A=B]\,ES = \{r = (w,s) : w \in ER \wedge s \in ES \wedge w[A] = s[B]\}$.

In the case A and B are the same attribute names, one of these two columns is deleted in the result. The join is called natural join (denoted by $\bowtie$). By analogy, the natural join of two relations ER and ES on $ATTR(ER) \cap ATTR(ES)$ can be formulated as follows:

$$ER \bowtie ES = \{r \mid (\exists w \in ER: w = r[ER] \wedge \exists s \in ES: s = r[ES])\} .$$

Example /see Figure 2 and 3/.

F-EMP ⋈ DEPT =

MNO	NAME	AGE	DNO	SAL	DNAME
100	Fischer	25	10	1000	A
101	Neuman	young	11	1500	B
102	King	young	11	high	B
103	Shmid	30	12	2000	C
104	John	old	13	very high	D

c. Section

Definition 3.4

Let ER be an extended relation over $ATTR(ER)$. A is an attribute and c is a value from $D(A) = U(A) \cup T(A)$. The selection A=c from ER (denoted by $ER\,[A = c\,]$) is defined by

$$ER[A=c] = \{r \in ER: (r[A]=c' \wedge c' \in U(A) \wedge c \in U(A) \wedge c=c')$$
$$\vee (r[A]=\underset{\sim}{u} \wedge \underset{\sim}{u} \in T(A) \wedge c \in U(A) \wedge \mu_{\underset{\sim}{u}}(c)=1)$$
$$\vee (r[A]=u_i \wedge u_i \in U(A) \wedge c \in T(A) \wedge \mu_c(u_i) \geq \tau)$$
$$\vee (r[A]=\underset{\sim}{u} \wedge \underset{\sim}{u} \in T(A) \wedge c \in T(A) \wedge (\exists u_i \in U(A):$$
$$\mu_{\underset{\sim}{u} \cap c}(u_i) \geq \tau \wedge \tau \in [0,1]))\}.$$

The value $\tau \in [0,1]$ is given by the user. If the user doesn't specify then $\tau = 0.5$ is assumed as default value.

We shall use F-EMP relation in Fig. 3 on an example.

Example

If we define $young(x) = \begin{cases} 1 & \text{for } x \leq 24 \\ 0.5 & \text{for } 25 \leq x \leq 30, \\ 0 & \text{otherwise} \end{cases}$ then

F-EMP [AGE = young] =

MNO	NAME	AGE	DNO	SAL
100	Fischer	25	10	1000
101	Neuman	young	11	1500
102	King	young	11	high
103	Shmid	30	12	2000

Remark

The remaining operations as UNION, INTERSECTION, DIFFERENCE for the extended relations are computed as in the theory of ordinary sets.

3.4 Fuzzy database and fuzzy relational algebra

Wirh the extended relational operations in the subsection 3.3 we have a possibility to process the data of a system more easily and finely. Therefore user's requirements can be satisfies more efficiently. In order to obtain an efficient database management system and an easy data processing system at high level the fuzzy data must be considered as the fuzzy sets. The stored data in Fig. 3 are neither "certain" nor "fuzzy term" data, but they are expressed in form of fuzzy sets. The information about every entity of a mini- world can be expressed simply by fuzzy sets and then more effectively processed.

In this case, any non-fuzzy value of a basic set /universe of discourse/ is assigned by a grade of membership and has a form $(1/u_i)$. Every value or fuzzy term of Fig. 3 is considered as a unique name of a corresponding fuzzy set. In /M. Umano, 79/ the author had given a method for data storage and processing in fuzzy database. The users requests are processed by 52 fuzzy sets operations. In this subsection we introduce a new method for processing of fuzzy data and fuzzy queries by fuzzy relation algebra. When no confusion occurs, $\underset{\sim}{U}$ is used to denote set of all fuzzy sets over a basic set U of an attribute. A fuzzy relation over an attribute set $\{A_1,\ldots,A_n\}$ /denoted by $FR(A_1,\ldots,A_n)$ with $U_i = U(A_i)$ and a fuzzy database are defined as follows.

Definition 3.5

A fuzzy database D_F is a collection of fuzzy relations

$$D_F = \{FR_1,\ldots,FR_n\},$$

where every fuzzy relation of fuzzy sets $\underset{\sim}{U}_1,\ldots,\underset{\sim}{U}_{n_i}$ in $U_1,\ldots,U_{n_i}$ is defined by a membership function

$$\mu_{FR}: \underset{\sim}{U}_1 \times\ldots\times \underset{\sim}{U}_{n_i} \to [0,1].$$

In other words, a fuzzy relation of the sets $\underset{\sim}{U}_1,\ldots,\underset{\sim}{U}_{n_i}$ is a subset of Cartesian product

$$FR \subseteq \underset{\sim}{U}_1 \times\ldots\times \underset{\sim}{U}_{n_i}$$

where U_i is determined on the universe of discourse of attribute A_i.

Remark

The Cartesian product was defined by /L.A. Zadeh, 75/

$$\mu: \underset{\sim}{U}_1 \times\ldots\times \underset{\sim}{U}_{n_i} \to [0,1]$$

where $\mu_{U_1 \times \ldots \times U_{n_i}}(r) = \bigwedge_{j=1}^{n_i} \mu_{U_j}(r_j)$, $r = (r_1,\ldots,r_{n_i}) \in \underset{\sim}{U}_1 \times \ldots \times \underset{\sim}{U}_{n_i}$, $r_j \in \underset{\sim}{U}_j$.

To illustrate a fuzzy relation of a fuzzy database in table form we add a special attribute name (in the sense of M. Umano, 79/. Below we introduce some fuzzy relational operations.

a. Fuzzy projection

Definition 3.6

Let FR be a fuzzy relation in ATTR(FR). A fuzzy projection of a fuzzy relation FR over a subset $X = \{A_1,\ldots,A_k\} \subseteq ATTR(FR)$ /denoted by $\pi_X(FR)$) is defined by

$$\pi_X(FR) = FR[X] = \{ (\bigvee_{\bar{X}} \mu_{FR}(r)/r(A_1,\ldots,A_k)) = \bigvee_{\bar{X}} \mu_{FR}(r)/(r[A_1],\ldots,r[A_k])$$

$$: \bar{X} = ATTR(FR) \setminus X \wedge A_i \in X \wedge \mu_{FR}(r)/r \in FR\}.$$

The fuzzy projection of a fuzzy relation is defined similarly as a projection of an extended relation. The duplicate tuples are identified and assigned the maximum μ- value.

We illustrate this Definition 3.6 with the following simple example.

Example

We consider two fuzzy relations

FR1(μ_1	A_1	A_2	A_3)
0.5	a	A	1
0.7	a	A	2
1.0	b	B	1
0.6	b	A	2

Fig. 4: FR1-fuzzy relation

FR2(μ_2	A_1	A_2	A_4)
0.6	a	A	x
0.8	a	A	y
0.9	b	A	y
0.9	b	B	x

Fig. 5: FR2-fuzzy relation

Let $X = \{A_1, A_2\} \subseteq ATTR(FR_1)$

$\pi_X(FR1) =$

(μ_1	A_1	A_2)
0.7	a	A
0.6	b	A
1.0	b	B

b. Fuzzy join

The definition of fuzzy join of the fuzzy relations is represented as follows.

Definition 3.7

Let FR and FS be two fuzzy relations in attribute set ATTR(FR) and ATTR(FS), respectively. $A \in ATTR(FR)$ and $B \in ATTR(FS)$ are two comparable attributes. A fuzzy equi-join of FR and FS on A and B is defined by

$$FR[A=B]FS = \{\mu/(w,s) : w[A] = s[B] \wedge \mu_{FR}(w)/w \in FR \wedge \mu_{FS}(s)/s \in FS \wedge \mu = \mu_{FR}(w) \wedge \mu_{FS}(s)\}.$$

The fuzzy sets w[A] and s[B] are compared by the rules in Definition 2.2. If A and B are the some attribute names then either A or B is deleted from result. This join is called natural and denoted by * . The Definition 3.7 can be generalized with the same way for a fuzzy join on a common attribute set ATTR(FR)∩ATTR(FS).

Example

/See Fig. 4 and 5/.

FR1 * FR2 =

(μ	A_1	A_2	A_3	A_4)
0.5	a	A	1	x
0.5	a	A	1	y
0.6	a	A	2	x
0.7	a	A	2	y
0.9	b	B	1	x
0.6	b	A	2	y

c. Fuzzy selection

Based on Definition 2.2, a fuzzy selection from a fuzzy relation is defined as follows.

Definition 3.8

Let FR be a fuzzy relation, A be an attribute of ATTR(FR) and c be an element of $\underset{\sim}{U}(A)$. A fuzzy selection A = c (denoted by $\sigma_{A=c}(FR)$) of relation FR is defined by

$$\tilde{\sigma}_{A=c}(FR) = \{\mu_{FR}(r)/r: \mu_{FR}(r)/r \in FR \wedge c \in \tilde{U}(A) \wedge$$

$$(\exists u_i \in U(A): \mu_{c \wedge r[A]}(u_i) \geq \tau \wedge \tau \in [0,1])\}.$$

Example (see Fig. 4)

$$\tilde{\sigma}_{A_1=a}(FR1) = \begin{array}{cccc} (\mu_1 & A_1 & A_2 & A_3) \\ 0.5 & a & A & 1 \\ 0.7 & a & A & 2 \end{array}$$

There are two tuples satisfy the condition $A_1=a$.

Remark

Other set-operations of the fuzzy relations can be defined based on the Definition 2.2. In this case it must be assumed that the fuzzy relations must have the same attribute set and every fuzzy relation is considered as a fuzzy set (level-2 fuzzy set).

Let FR1 and FR2 be two fuzzy relations on $\tilde{U} = \tilde{U}_1 \times \ldots \times \tilde{U}_n$ where

$$FRj = \{\mu_{FRj}(u_1,\ldots,u_n)/(\tilde{u}_1,\ldots,\tilde{u}_n): \tilde{u}_i \in \tilde{U}_i, \quad i=1,\ldots,n\}$$

$$= \{\mu_{FRj}(\tilde{u})/\tilde{u}: \tilde{u} \in \tilde{U}\}, \quad j=1,2.$$

Briefly we represent this set operation in the following Fig. 6.

Name	Notation	Representation
fuzzy union	FR1∪FR2	$FR1 \cup FR2 = \{\mu_{FR1}(\underset{\sim}{u}) \vee \mu_{FR2}(\underset{\sim}{u}) / \underset{\sim}{u} : \underset{\sim}{u} \in \underset{\sim}{U}\}$
fuzzy intersection	FR1∩FR2	$FR1 \cap FR2 = \{\mu_{FR1}(\underset{\sim}{u}) \wedge \mu_{FR2}(\underset{\sim}{u}) / \underset{\sim}{u} : \underset{\sim}{u} \in \underset{\sim}{U}\}$
difference (minus)	FR1⊖FR2	$FR1 \ominus FR2 = \{(\mu_{FR1}(\underset{\sim}{u}) - \mu_{FR2}(\underset{\sim}{u})) \vee 0 / \underset{\sim}{u} : \underset{\sim}{u} \in \underset{\sim}{U}\}$
Cartesian product	FR1⊗FR2	$FR1 \otimes FR2 = \{\mu_{FR1}(\underset{\sim}{u}) \wedge \mu_{FR2}(\underset{\sim}{v}) / (\underset{\sim}{u}\ \underset{\sim}{v}) : \underset{\sim}{u} \in \underset{\sim}{U},\ \underset{\sim}{v} \in \underset{\sim}{V}\}$ $= \{\mu_{FR1}(\underset{\sim}{u}_1, \ldots, \underset{\sim}{u}_n) \wedge \mu_{FR2}(\underset{\sim}{v}_1, \ldots, \underset{\sim}{v}_n) / (\underset{\sim}{u}_1, \ldots, \underset{\sim}{v}_n) : \underset{\sim}{u}_i \in \underset{\sim}{U}_i\ \ \underset{\sim}{v}_j \in \underset{\sim}{V}_j\}$

Fig. 6: Set operations on fuzzy relation

4. APPLICATIONS OF FUZZY RELATIONAL EXPRESSIONS TO FUZZY RELATIONS BY EXAMPLES OF SEQUEL-2

In this section we shall formulate some applications of fuzzy relational expressions to fuzzy relations which is an extended version of Codd's model.

The retrieval from fuzzy database is performed on the basis of relational algebraic operations. In the general case, the representation of some user's request in the form of a relational expression is difficult. Here we would like to limit our observation only to a queries class which can be formulated nonprocedurally in terms of fuzzy relational expression, namely in terms of the SEQUEL-language /see D.D. Chamberlin et al. 78/.

Retrieval method

The user's request is formulated in the form of a fuzzy relational expression. (i.e. of the form {t| (t)}, see Ullman/.

Any predicate /atom/ of a fuzzy relational expression contains fuzzy sets as constants /with their unique names in the representation/ and fuzzy equality as predicate operator.

The predicates are evaluated with respect to a tuple of a fuzzy relation by a two-valued logic. If a predicate satisfies the conditions, then it obtains a truth-value 1. If it doesn't, there it obtains a truth-value 0.

The predicates are connected by the Boolean operations OR, AND and NOT.

As for evaluation of fuzzy relational expression, the predicates are evaluated ranging over all tuples of the relations and if the predicates are true then the tuple constucted from the components corresponding to target list of the expression.

The grade of membership of any tuple in a fuzzy relation is computed and processed basing on Definition 3.6-Definition 3.8.

Remark

The predicates introduced here are evaluated by a two-valued logic. In the general case, they could be considered as fuzzy predicates which are evaluated not only by two-valued logic but also by fuzzy logic /see Le Tien Vuong and Ho/.

Example

Here we shall consider as example a fuzzy database which consists of two fuzzy relations F-EMP and F-DEPT /see fig. 7 and 8/.

F-EMP(μ	MNO	NAME	AGE	DNO	SAL
0.8	101	A	20	10	very high
0.9	102	B	25	11	high
0.8	103	A	more or less 20	10	1000
0.9	104	D	young	12	2000
0.8	105	D	old	12	1500
1.0	106	B	50	11	500

Fig. 7: F-EMP relation

F-DEPT (μ	DNO	DNAME	LOC
1.0	10	x	London
1.0	11	y	London
1.0	12	z	Manchester
1.0	13	w	Liverpool

Fig. 8: F-DEPT relation

Relation F-EMP describes a set of employees, giving the employee person ordinal number, name, age, department ordinal number and salary for each employee. The F-DEPT relation consists of department ordinal number, name and location of each department. The attribute name shows the grade of membership of each relation tuple.

Some fuzzy sets in Fig. 7 and 8 can be defined basing upon the techniques developed by /L.A. Zadeh, 75/ as follows.

more or less 20 := {0.5/19, 1.0/20, 0.6/21},

$$\text{young}(x) = \begin{cases} 1 \text{ for } x \leq 24 \\ 0.5 \text{ for } 25 \leq x \leq 30 \\ 0 \text{ otherwise} \end{cases}$$

$$\text{old}(x) = \begin{cases} 1 \text{ for } x \geq 60 \\ 0.5 \text{ for } 55 \leq x < 60 \\ 0 \text{ otherwise} \end{cases}$$

$$\text{high}(x) = \begin{cases} 1 \text{ for } x \geq 1800 \\ 0.6 \text{ for } 1500 \leq x < 1800 \\ 0 \text{ otherwise} \end{cases}$$

$$\text{very high}(x) = [\text{high}(x)]^2.$$

The following examples will be represented in terms of SEQUEL 2.

Q1. List of the names of employees who are 20 yr old.

```
SELECT UNIQUE NAME
FROM F-EMP
WHERE AGE = 20
```

The selection AGE = 20 is first executed from F-EMP relation. The following tuples satisfy this condition

```
(0.8  101 A    20           10 very high)
(0.8  103 A  more or less 20  10  1000)
(0.9  104 D  young           12      2000).
```

Then the projection on the NAME attribute is executed with eliminating the duplicate tuples. The last result is

{0.8/A , 0.9/D}.

Q2. List of names and employee person ordinal number who young and have a high salary:

```
SELECT MNO, NAME
FROM F-EMP
WHERE AGE = young AND SAL = high
```

The first, second and fourth tuple of F-EMP relation are satisfy the selection conditions. .
After the projection is executed on NAME and MNO we have the result:

{0.8/(101,A), 0.9/(102,B), 0.9/(104,D)}

Q3. List the departments which have no employees.

```
SELECT DNO
FROM F-DEPT
MINUS
SELECT DNO
FROM F-EMP
```

The first projection is performed on attribute DNO from F-DEPT relation. The result is {1.0/10, 1.0/11, 1.0/12, 1.0/13}. The second projection is performed on DNO from F-EMP relation an we obtain following tuples {0.8/10, 1.0/11, 0.9/12}. Now the minus operations /see Fig. 6/ are executed and we have the following result /for τ= 0.5/

{1.0/13}.

Q4. List the names of all employees and the locations where they work.

```
SELECT UNIQUE F-EMP.NAME,F-DEPT.LOC
FROM F-EMP,F-DEPT
WHERE F-EMP.DNO = F-DEPT.DNO
```

The join operation is first executed on attribute DNO of F-EMP relation and F-DEPT relation. We have the following result

```
(0.8  101  A       20          10 very hight  x  London)
(0.8  103  A  more or less 20  10       1000   x London)
(0.9  102  B       25          11       high   y London)
(1.0  106  B       50          11        500   y London)
(0.9  104  D    young          12       2000   z Manchester)
(0.8  105  D    old            12       1500   z Manchester)
```

The projection (with eliminating of duplicate tuples) is executed on F-EMP.NAME and F-DEPT.LOC. We have the following result

{0.8/(A,London), 1.0/(B,London), 0.9/ (D,Manchester)}

5. CONCLUSION

In this paper the Codd's relational model of data is extended using fuzzy sets theory. We propose fuzzy relational model in which fuzzy data are represented by fuzzy relations.

We have introduced some fuzzy relational operations, namely, projection, join, selection, union, intersection, difference... . The retrieval method for such a fuzzy database is presented in detail by several examples. This method is based on fuzzy relational algebra. By generalizing appropriately comparison operations between fuzzy sets, we hope that the processing capacities of a relational fuzzy database can be more powerful. That is the aim of our subsequent paper.

ACKNOWLEDGEMENT

The authors wish to thank Prof. N.D. Ngoc for his valuable remarks and suggestions.

REFERENCES

BELKE 78 W. BELKE, et...: Zur Nutzung unscharfer Mengen im formal System der Klassifizierung. TU-Infor. 08-02-78, Dresden.

CODD 70 E.F. Codd: A relational model of data for large shared data banks, C.ACM 13,6(1970), 377-387.

CHAMBERLIN et al. 78: D.D. Chamberlin et al.: SEQUEL-2: A unified approach to data definition, manipulation and control, IBM-Jour., Res. and Dec, 20,6(1978) 560-575.

LE 83 Le Tien Vuong: Untersuchung zur ternaeren Dekomposition einer Relation und zur Anwendung der unscharfen Menge in CRM, Diss. TU-Dresden (1983).

LE 85 Le Tien Vuong: On the applications of fuzzy sets theory to relational database, J. Computer Sc., and Cyb. Vol. 1, N.4 (1985) (in vietnamese).

LE and HO in preparation

NEGOITA et al. 75: C.V. Negoita, et al.: Applications of fuzzy sets to systems analysis, Birkhhauser Verlag, Basel und Stuttgart, 1975.

TAHANI 77 V. Tahani: A conceptual framework for fuzzy queries processing- a step toward very intelligent database system, Inf.Proc. Manag. 13(1977) 289-303.

UMANO 79 M. Umano: Representation and manipulation of fuzzy data, Diss., Osaka Univ. Japan 2. 1979.

ZADEH 75 L.A. Zadeh: The concept of a linguistic variable and its application to approximate reasoning, Inf. Sc. Vol. 8, 199-248, 301-357 (1975).

ULLMAN Principles of database systems, second edition, Computer Science Press (1982).

Retrieval from fuzzy database by fuzzy relational algebra

Le Tien Vuong, Ho Thuan

Summary

In this paper a fuzzy relational model for fuzzy data - considered as an extended version of Codd's model-is developed.

A method for fuzzy data and queries processing is introduced, based on the fuzzy relational operations, namely, projection, join, selection, union, intersection.

These are illustrated by several examples and their applications to fuzzy database are presented in detail by the sublanguage SEQUEL-2.

Fuzzi adathalmazból való visszakeresés fuzzi relációs algebra segitségével

Le Tien Vuong, Ho Thuan

Összefoglaló

A cikkben a szerzők a Codd-féle relációs modellnek egy kiterjesztését ismertetik, amelyben "fazzi" /"fuzzy"/ adatok is szerepelhetnek. A kiterjesztés az un. fazzi /"fuzzy"/ relációs adatmodell. Bevezetnek több "fazzi" relációs müveletet /vetités, összekapcsolás, kiválasztás, unió, metszet/. A mondottakat több egyszerü példán szemléltetik, valamint az un. SEQUEL-2 nyelv segitségével is leirják ezeket a példákat.